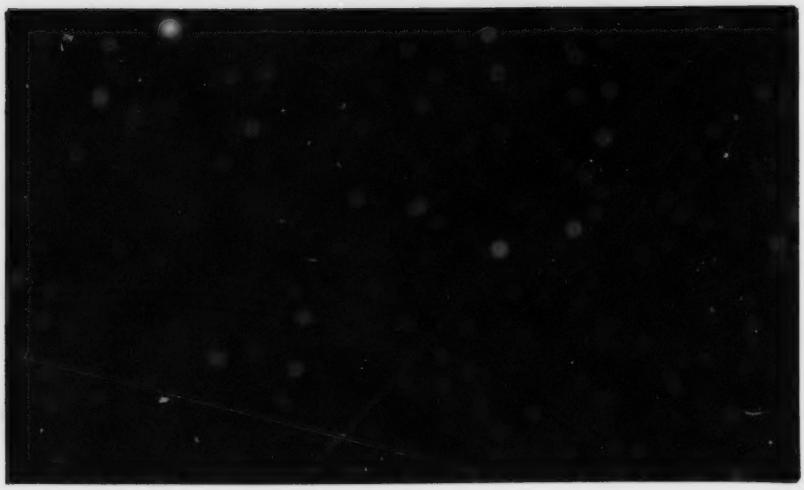
CENTRE
DE RECHERCHE
SUR LES TRANSPORTS
(CRT)
CENTRE
FOR RESEARCH
ON TRANSPORTATION

UNIVERSITÉ DE MONTRÉAL
ÉCOLE DES HAUTES ÉTUDES COMMERCIALES
ÉCOLE POLYTECHNIQUE



THÈSE DE DOCTORAT

C.P. 6128, SUCCURSALE A
MONTRÉAL (QUÉBEC)
CANADA
H3C 3J7

TÉLÉPHONE : (514) 343-7575
TÉLEX : 05-24146 BIBPOLYTEC
TÉLÉCOPIEUR (FAX) : (514) 343-7121

CRT-671

CONFECTION D'ITINÉRAIRES POUR LE TRANSPORT ADAPTÉ*

par

Yvan Dumas

* Thèse de doctorat, département de Mathématiques appliquées, École Polytechnique de Montréal, dirigée par Messieurs François Soumis et Jacques Desrosiers.

Centre de recherche sur les transports - Publication #671

Décembre 1989

90-03009/

Dépôt légal - 4e trimestre 1989

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE

Cette thèse intitulée:

CONFECTION D'ITINÉRAIRES
POUR LE TRANSPORT ADAPTÉ

présentée par: Yvan Dumas
en vue de l'obtention du grade de: Philosophiae Doctor
a été dûment acceptée par le jury d'examen constitué de:

Monsieur J. Gauvin, Ph.D., président
École Polytechnique de Montréal
Département de Mathématiques appliquées

Monsieur F. Soumis, Ph.D., directeur
École Polytechnique de Montréal
Département de Mathématiques appliquées

Monsieur J. Desrosiers, Ph.D., co-directeur
École des Hautes Études Commerciales de Montréal
Service des Méthodes Quantitatives et des Système d'Information

Monsieur J.M. Rousseau, Ph.D., examinateur
Université de Montréal
Département d'Informatique et de Recherche Opérationnelle

Monsieur A.R. Odoni, Ph.D., examinateur
Massusetts Institute of Technology
Center for Transportation Studies

À Michèle ainsi que
Caroline et Phillipe.

SOMMAIRE

Cette thèse présente un ensemble d'algorithmes permettant de confectionner les itinéraires et l'horaire de véhicules en vue du transport de personnes handicapées. Le problème à résoudre est celui de satisfaire, aux moindres coûts, un ensemble de requêtes de déplacement dont chacune d'elle exige l'embarquement à un lieu spécifique et le débarquement à un autre lieu spécifique. Le scénario d'opération suppose: les requêtes connues à l'avance; un service porte à porte; des fenêtres de temps pour les embarquements et les débarquements afin d'assurer un niveau de service acceptable; des requêtes pouvant consommer des ressources différentes (exigeant ainsi des véhicules des capacités multidimensionnelles) et finalement des véhicules de caractéristiques différentes.

Les contributions majeures de cette thèse sont: (1) le concept innovateur de tâche ("mini-cluster"), (2) la généralisation d'un algorithme de génération de colonnes pour le problème de voyageur de commerce multiple avec fenêtres horaires pour résoudre les problèmes avec plusieurs dépots et des véhicules de caractéristiques différentes, (3) une méthode de décomposition géographique et temporelle permettant de résoudre de très gros problèmes de voyageur de commerce multiple avec fenêtres horaires, (4) un algorithme exact de programmation dynamique permettant de résoudre de beaucoup plus gros problèmes de voyageur de commerce avec contraintes additionnelles de temps, de capacités et de priorité, (5) un algorithme dual, de complexité linéaire, pour confectionner l'horaire d'un itinéraire sous des fonctions d'inconvénient convexes sur les heures de passage et, finallement, (6) une procédure d'estimation discrète des temps de déplacement.

Les problèmes que nous voulons résoudre sont très complexes et de très grandes tailles. Puisque la technologie présente ne permet pas de les résoudre de façon optimale, nous proposons une approche de résolution hybride formée d'heuristiques et de méthodes mathématiques d'optimisation. A notre connaissance, il n'existe que deux autres procédures ayant démontrées leur capacité à résoudre des problèmes de très grande taille. Notre approche se distingue de celles-ci entre autres par l'utilisation de méthodes mathématiques d'optimisation qui aident à conserver une vue d'ensemble du problème, fournissant ainsi de meilleurs

résultats.

L'approche proposée se divise en trois phases. La première répartit les requêtes entre les véhicules (**phase de répartition** ou "clustering"). La deuxième établit les itinéraires de chacun des véhicules (**phase d'ordonnancement** ou "routing") et la troisième détermine l'horaire de chacun des itinéraires (**phase de confection d'horaire** ou "scheduling"). Par contre, les trois niveaux de décisions ne sont pas pris de façon disjointe. Pour assurer la faisabilité des phases aux niveaux inférieurs, il doit y avoir intéraction entre les différents types de décisions.

La première phase, effectuant la répartition des requêtes entre les véhicules, se divise en 2 étapes. Dans la première étape intervient l'important concept de tâche. Une procédure prend des décisions locales sur le regroupement et l'ordonnancement des requêtes de façon à regrouper dans des portions d'itinéraires faisables des requêtes voisines (de 3 à 4 requêtes en moyenne). Chacune des portions d'itinéraire forme ainsi une tâche. Les propriétés recherchées de ces tâches sont: (1) la faisabilité: la portion d'itinéraire de chaque tâche doit satisfaire l'ensemble des requêtes de la tâche en respectant les contraintes de fenêtres de temps, de couplage, de priorité et de capacité du véhicule, (2) l'efficacité: la somme des temps de déplacement des véhicules à l'intérieur des tâches est assez proche du minimum. Plusieurs procédures heuristiques peuvent être utilisées pour créer un ensemble de tâches satisfaisant les propriétés ci-haut mentionnées. Par exemple, les requêtes pourraient être tout simplement regroupées selon des critères de proximité et de faisabilité. Nous avons plutôt utilisé une procédure qui effectue le morcellement des itinéraires d'une solution initiale aux endroits où le véhicule se déplace sans passager. Il est facile de se convaincre que les tâches ainsi créées ont la propriété de faisabilité. Cette solution initiale peut aussi bien être une solution manuelle, une solution générée par un des deux algorithmes déjà existant ou, après une itération, la solution produite par notre approche (ré-optimisation).

Dans la deuxième étape de la première phase, on répartit et ordonne les tâches entre les véhicules en solutionnant un problème de voyageur de commerce multiple avec fenêtres de temps (m-TSPTW) et contraintes de ressources. A ce niveau les décisions globales sur la répartition sont effectuées en prenant en

considération les coûts de répartition, les coûts d'ordonnancement ainsi que la faisabilité des phases suivantes. Une approche optimale pour le m-TSPTW avec un seul type de véhicules a été proposée; cette approche peut traiter rapidement des problèmes d'une centaine de tâches. L'approche est généralisée au chapitre 2 pour considérer une flotte mixte de véhicules avec plusieurs dépots ainsi que des contraintes de compatibilité entre les véhicules et les tâches. Le nombre de tâches estimé étant de l'ordre de 500 à 1000 pour des problèmes de 2000 requêtes, il est impossible de penser résoudre le m-TSPTW directement. Ainsi, nous présentons au chapitre 5 une décomposition géographique et temporelle du problème qui permet d'obtenir une suite de m-TSPTW de taille raisonnable à résoudre. Cette approche génère des itinéraires sur les tâches et assure l'existence d'itinéraires et d'horaires faisables (sur les requêtes) pour chaque véhicule. Toutefois, un itinéraire efficace sur les tâches n'implique pas un itinéraire efficace sur les requêtes.

Dans la deuxième phase, on réordonne optimalement les requêtes de chaque véhicule en solutionnant un problème de voyageur de commerce avec contraintes additionnelles de temps, de priorité et de capacité (1-PDPTW). Les algorithmes exacts proposés jusqu'à présent n'ont permis de résoudre que de petits problèmes—moins de 15 requêtes. Aucun n'a exploité la contrainte exigeant que les requêtes affectées à un véhicule doivent être toutes desservies par celui-ci. Cette contrainte est très restrictive et elle permet d'élaborer plusieurs critères efficaces d'élimination d'états (dans le contexte de la programmation dynamique). Au chapitre 3, nous proposons un algorithme exact de programmation dynamique, avec l'élaboration de plusieurs critères d'élimination d'états, qui permet de résoudre plus efficacement des problèmes de grande taille—plus de 40 requêtes.

Les itinéraires construits à partir du regroupement des requêtes en tâches, de l'affectation des tâches aux véhicules et de la réoptimisation de l'ordonnancement des arrêts sont faisables. La procédure est complétée dans une troisième phase par l'utilisation d'un algorithme exact déterminant les heures de services aux arrêts de l'itinéraire. Il y a généralement plusieurs horaires qui sont compatibles avec l'itinéraire et les fenêtres horaires. Le temps d'attente du véhicule peut être réparti de plusieurs façons le long de cet itinéraire. Le problème à résoudre est celui de choisir l'horaire qui minimise les inconvénients associés aux fenêtres horaires à

chacun des noeuds. Les inconvénients sont modélisés par des fonctions convexes sur les heures de service. Les fonctions convexes permettent de générer des solutions moins extrémistes que les fonctions linéaires considérées dans la littérature. L'algorithme proposé est présenté au chapitre 4.

Afin de réaliser les tests numériques, nous avons dû développer une fonction d'estimation des temps de déplacement fournissant des temps aux 5 minutes comparables à ceux évalués par les répartiteurs. L'annexe A présente des modèles statistiques d'estimation où la variable dépendante (le temps de déplacement) est une variable discrète. Un algorithme original permet de résoudre ce problème mathématique. Notre fonction d'estimation ainsi développée permet d'estimer de façon exacte les valeurs observées dans une proportion de 83% et d'engendrer moins de 0.5% d'erreurs de plus de 5 minutes.

Finalement, l'approche complète est testée sur des données disponibles provenant d'une journée réelle d'opération à la Wheel-Trans Service de Toronto. Le problème contient 1890 requêtes ainsi qu'un ensemble de 133 pièces de travail. Une nouvelle fonction économique est également présentée pour favoriser les solutions permettant une insertion de requêtes ultérieures.

MOTS CLES : transport pour handicapés, routage, horaire, génération de colonnes, "clustering".

ABSTRACT

This thesis present a set of algorithms to construct vehicle routes in a large scale multi-vehicle many-to-many system in the context of the transportation of the handicapped. The problem consist in minimizing operation costs to satisfy a set of transportation requests, each one involving pickup and delevery stop. The scenario assumed is one of: advance requests; point-to-point transportation; hard time windows to ensure acceptable levels of service; heterogeneous vehicle fleet; and non-identical passenger characteristics with different types of passengers imposing different demands with regard to vehicle capacity.

The main contributions of this thesis are (1) the innovative concept of mini-cluster, (2) the generalization of the column generation algorithm for the Multiple Travelling Salesman Problem with Time Windows (m-TSPTW): the generalization concerns multiple depots and heterogeneous vehicle fleet, (3) a space/time decomposition method, with global seeing, to solve very large m-TSPTW, (4) an exact dynamic programming algorithm to solve larger size Travelling Salesman Problem with additional time window, capacity and precedence constraints, (5) a dual-based, time schedule optimization approach to schedule a fixed path with time window and convex inconvenient function cost on visiting time, (6) a discret estimation procedure to estimate travelling time.

The problems to be solved are very large and complex. The current technology does not permit to optimally solve problems of realistic size which present this complexity. Instead, we present a procedure which use a hybrid of heuristic methods and mathematical optimization. To our knowledge, there is only two other algorithms that have been tested with problems of very large size. Our approach differ from those by the use of mathematical optimization which help us to keep global seeing of the problem.

The proposed approach is divided in three phases. However, the different levels of decision in the phases are not taken in the disjoint manner to ensure faisability of inferior phases. In the first phase (**clustering phase**), the original set of requests is partitioned into clusters, each of them is to be served by a single vehicle. In the second phase (**routing phase**), routing is carried out for each

vehicle in a separate step. In the last phase (**scheduling phase**), the schedule of each vehicle is optimized over a fixed itinerary.

The **clustering phase** is itself divided in two stages. At the first stage, we use the innovative concept of mini-clusters. We only focus on local temporal and spatial considerations to group requests into mini-clusters. We group requests into mini-clusters corresponding to small work periods for one vehicle rather than grouping requests into clusters corresponding to a full day's work for one vehicle. Each mini-cluster groups together requests that are in about the same time and the same direction and that can therefore be served with the same vehicle. One feature of this approach is the use of an existing feasible solution in the creation of mini-clusters. Thus, each mini-cluster actually represent a small portion of a feasible vehicle route.

At the second stage of the clustering stage, routes for all the vehicles are constructed simultaneously in solving a m-TSPTW with ressource and depot constraints. In this way, clusters are created and the faisability of the next two phases is ensured. This step deals with global operational planning considerations. An approach have been presented to solve the m-TSPTW with an homogeneous fleet. This approach rapidly solve problems with 100 tasks (mini-clusters, in our context). The approach is generalized in chapter 2 to consider an heteregeneous vehicle fleet, multiple depots and compatibility constraints between tasks and vehicles. As the estimate number of mini-clusters is between 500 and 1000 for a 2000 requests size problem, it is not possible to solve it directly. Chapter 5 present a space/time decomposition method to be able to solve very large m-TSPTW using the exact approach presented in chapter 2.

In the **routing phase**, mini-clusters are reopened and each vehicle route is reoptimized, now taking into account the actual arrangement of the mini-clusters from the clustering phase. At this point the nodes visited by a vehicle may be reordered. This phase is carried out optimaly using a dynamic programming algorithm. The problem to be solved is a Travelling Salesman Problem with additional time window, precedence and capacity constraints (1-PDPTW). In the litterature, only small problems (less than 15 requests) have been solved. An important con-

straint has not been considered: the constraint to visit all the requests with the same vehicle. This constraint permit an efficiente elimination of states in the dynamic programming context. In chapter 3, we propose an exact algorithm using dynamic programming which easily solves problems with up to 40 requests.

In the **scheduling phase**, the schedule of each vehicle is optimized over a fixed itinerary. There are generally several schedules which are compatible with the chosen itinerary and the time windows available for visiting each node. Vehicle idle time may generally be distributed in a number of ways over the course of the itinerary. The problem is one of choosing the schedule which minimize the inconvenience associated with service time at each of the nodes. At each node, inconvenience is modelled using a convex function of the service time. This is more general than the linear functions used up to now in the literature. An optimal dual method for solving this problem is presented in chapter 4.

To realize the numerical tests, we needed a function to estimate, by step of 5 minutes, travelling times given a sample of estimated travelling times proposed by the dispatchers. The annex A present some statistic models where the dependant variable is of discret type. A nice algorithm is shown to solve this mathematical problem.

Finally, the overall approach is tested on data from a real operation day at the Wheel-Trans Service of Toronto. The problem includes 1890 requests and 133 pieces of work. A new objective function is also shown to generate solutions which permit the insertion of further requests.

KEY WORDS : dial-a-ride, routing, scheduling, column generation, clustering.

REMERCIEMENTS

Une thèse de cette envergure ne se réalise pas seul en face d'une table, de feuilles et d'un terminal. Elle s'est réalisée au sein d'une équipe dynamique, l'équipe D^3S^2. Cette équipe est composée de François Soumis (directeur), Jacques Desrosiers (co-directeur), Michel Sauvé, Martin Desrochers ainsi que moi-même. Je leur exprime ma profonde gratitude.

Je voudrais remercier tout particulièrement mes directeurs François Soumis et Jacques Desrosiers pour leur soutient financier, leur disponibilité, leur supervision et pour les nombreuses heures constructives et productives passées à discuter avec moi.

Je remercie la Wheel Trans Service de Toronto et la compagnie GIRO inc. d'avoir bien voulu nous fournir l'ensemble des données d'une journée d'opération.

Je remercie le Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG) ainsi que le Fonds pour la formation de chercheurs et l'aide à la recherche (FCAR) pour l'aide financière accordée.

Enfin, je remercie énormément mon épouse Michèle pour son soutien moral et son temps passé à corriger mes chapitres dont elle juge le contenu "incompréhensible". Et je m'engage désormais auprès de Michèle, Caroline (2 ans) et Philippe (2 mois) à leur accorder toute l'attention requise, ce qui fut parfois négligée durant ma période d'études.

TABLE DES MATIÈRES

SOMMAIRE v

ABSTRACT ix

REMERCIEMENTS xii

LISTE DES FIGURES xvii

LISTE DES TABLEAUX xix

CHAPITRE 1. INTRODUCTION 1

1.1 Le problème du transport des personnes handicapées 2

1.2 Scénario d'opération 3

1.3 Formulation mathématique 6

1.4 Revue de la littérature 11

1.4.1 Le problème du voyageur de commerce avec fenêtres horaires (TSPTW) 12

1.4.1 Le problème du voyageur de commerce multiple avec fenêtres horaires (m-TSPTW) 13

1.4.3 Le problème du plus court chemin avec fenêtres horaires (SPPTW) 15

1.4.4 Le problème du transport de marchandise avec fenêtres horaires (VRPTW) 16

1.4.5 Le problème de cueillettes et livraisons avec fenêtres horaires (PDPTW) 17

1.5 Méthodologie et approche proposées 19

1.6 Références 24

CHAPITRE 2. RÉPARTITION DES REQUÊTES ENTRE LES VÉHICULES 31

2.1 Sommaire 32

2.2 The Multiple Vehicle DIAL-A-RIDE Problem, compte rendu de conférence publié dans *Computer-Aided Transit Scheduling* 308 . . 35

CHAPITRE 3. OPTIMISATION DE L'ITINÉRAIRE D'UN VÉHICULE 58

3.1 Sommaire 59

3.2 A Dynamic Programming Solution of the Large-Scale Single-Vehicle DIAL-A-RIDE Problem with Time Windows, article publié dans *American Journal of Mathematical and Management Sciences* 6 61

CHAPITRE 4. OPTIMISATION DE L'HORAIRE D'UN ITINÉRAIRE . 86

4.1 Introduction 87

4.2 Formulation mathématique 89

4.3 Pré-traitement 90

4.4 Réduction du problème 91

4.5 Approche de résolution 93

4.6 Algorithme 96

4.7 Spécialisation au cas de fonctions quadratiques 100

4.8 Spécialisation au cas de fonctions linéaires 101

4.9 Extensions 104

4.10 Conclusion 106

4.11 Références 107

CHAPITRE 5. EXPÉRIMENTATION DU PROTOTYPE DARSY . . . 110

5.1 Le prototype DARSY 111

5.1.1 Découpage temporel et géographique 115

5.2 Expérimentation 118

5.2.1 Les données 118

5.2.2 Plan d'expérience 122

5.2.3 Les solutions initiales 126

5.2.4 Analyse des résultats 131

5.3 Références 143

CONCLUSION 144

ANNEXE A. ESTIMATION DES TEMPS DE DÉPLACEMENT 147

A.1 Introduction 148

A.2 Description des données 148

A.3 Modèle de régression unidimensionnel 150

A.3.1 La méthode des moindres carrés 151

A.3.2 La méthode de la somme minimum des erreurs 152

A.3.3 La méthode du nombre minimum d'erreurs 152

A.3.4 Problèmes d'estimation 153

A.4 Modèle de régression bidimensionnel 155

A.5 Modèle de régression bidimensionnel avec périodes de pointe . . . 158

A.6 Conclusion 160

Appendices 169

A.A Démonstration des propositions 169

A.B Algorithme de résolution des problèmes d'estimation 176

LISTE DES FIGURES

CHAPITRE 3. OPTIMISATION DE L'ITINÉRAIRE D'UN VÉHICULE

Figure 1: Effect of Elimination Criterion # 9 on the Number of States Generated 74

Figure 2: Numerical Example 76

Figure 3: Execution Time as a Function of the Number of Requests . . 80

CHAPITRE 4. OPTIMISATION DE L'HORAIRE D'UN ITINÉRAIRE

Figure 1: Intervalle de temps et fonction d'inconvénient associés à chaque noeud 89

Figure 2: Structure du problème de flot à coût minimum 109

CHAPITRE 5. EXPÉRIMENTATION DU PROTOTYPE DARSY

Figure 1: Découpage temporel d'une période 116

Figure 2: Découpage géographique d'Ouest en Est puis d'Est en Ouest d'un schéma d'une ville 118

Figure 3: Description des requêtes 120

Figure 4: Ensemble des pièces de travail 121

Figure 5: Répartition des périodes libres de la solution TOR-5m . . . 134

Figure 6: Répartition des périodes libres de la solution C 135

ANNEXE A. ESTIMATION DES TEMPS DE DÉPLACEMENT

Figure 1: Distribution des temps d'inter-arrêts en fonction des distances corrigées 161

Figure 2: Répartition des temps d'inter-arrêts 161

Figure 3: Répartition des temps de service à l'adresse de départ 162

Figure 4: Courbes des fonctions unidimentionnelles $\tau(D)$ de la méthode des moindres carrés 162

Figure 5: Courbes des fonctions unidimentionnelles $\phi(D)$ de la méthode de la somme des erreurs absolues 163

Figure 6: Courbes des fonctions unidimentionnelles $\delta(D)$ de la méthode du nombre minimum d'erreurs 163

Figure 7: Intervalles de variation de la vitesse du modèle unidimentionnel pour lequel les paramètres sont estimés par le problème $(P1)$ 164

Figure 8: Courbe d'évaluation de l'effet de la période de la journée sur l'estimation des paramètres 164

LISTE DES TABLEAUX

CHAPITRE 2. RÉPARTITION DES REQUÊTES ENTRE LES VÉHICULES

Table 1: Test Problem Description 51

Table 2: Results of Problems Solved without Time Slice Decomposition at the Mini-Cluster Routing Stage 52

Table 3: Results of Problems Solved by Time Slice Decomposition at the Mini-Cluster Routing Stage 53

CHAPITRE 3. OPTIMISATION DE L'ITINÉRAIRE D'UN VÉHICULE

Table 1: Problem characteristics 79

Table 2: Summary of Results of 95 Problems 82

Table 3: Elimination Statistics by Criterion 82

CHAPITRE 4. OPTIMISATION DE L'HORAIRE D'UN ITINÉRAIRE

Tableau 1: Description des arcs du réseau 103

CHAPITRE 5. EXPÉRIMENTATION DU PROTOTYPE DARSY

Tableau 1: Procédure générale de DARSY 112

Tableau 2: Plan d'expérience 125

Tableau 3: Description des solutions initiales 127

Tableau 4: Description des tâches obtenues par morcellement des solutions initiales 130

Tableau 5: Comparaison des solutions TOR-5m, A, B et C 133

Tableau 6: Comparaison des solutions ROY-5m, D, E et F 140

Tableau 7: Comparaison des solutions ROY-1m, G, H et I 141

Tableau 8: Comparaison des solutions G, J, H et K 142

ANNEXE A. ESTIMATION DES TEMPS DE DÉPLACEMENT

Tableau 1A: Valeurs (en km) des paramètres estimés par le problème ($P1$) pour le modèle unidimentionnel 165

Tableau 1B: Distribution des erreurs $T_k - \widehat{T}(D_k)$ où $\widehat{T}(D)$ est définie avec les estimateurs du tableau 1A 165

Tableau 2A: Valeurs (en km) des paramètres estimés par le problème ($P2$) pour le modèle unidimentionnel 166

Tableau 2B: Distribution des erreurs $T_k - \widehat{T}(D_k)$ où $\widehat{T}(D)$ est définie avec les estimateurs du tableau 2A 166

Tableau 3A: Valeurs (en km) des paramètres estimés par le problème ($P2$) pour le modèle bidimentionnel 167

Tableau 3B: Distribution des erreurs $T_k - \widehat{T}(D_k, M_k)$ où $\widehat{T}(D, M)$ est définie avec les estimateurs du tableau 3A 167

Tableau 4A: Valeurs (en km) des paramètres estimés par le problème ($P2$), selon la méthode des moindres carrés, pour le modèle bidimentionnel en période de pointe et en période hors pointe 168

Tableau 4B: Distribution des erreurs $T_k - \widehat{T}(D_k, M_k)$ où $\widehat{T}(D, M)$ est définie avec les estimateurs du tableau 4A 168

1

Introduction

1.1 Le problème du transport des personnes handicapées

Depuis 1978, les organismes publics de transport en commun du Québec ont l'obligation, par le biais de la loi 9 assurant l'exercice des droits des personnes handicapées, d'offrir un service de transport adapté aux personnes handicapées résidant sur leur territoire. Actuellement, on dénombre plus de 60 organismes de transport adapté publics au Québec (Cloutier et Mongrain 1986) et plus de 20 000 personnes handicapées admises. Ces dernières ont effectué plus de un million de déplacements en 1985. Par exemple, pour cette même année, la société de transport de la communauté urbaine de Montréal effectuait plus de 1 400 déplacements durant une journée typique de semaine. Pour la même période, la Wheel-Trans Service de Toronto en effectuait plus de 2100.

Le nombre et la taille de tels services ne cessent d'augmenter. On prévoit que, d'ici 20 ans, la demande de déplacements en transport adapté augmentera d'au moins 800% (Bélisle 1986). L'on tient compte, ici, du fait que le nombre de personnes à mobilité réduite, la proportion de celles-ci admises au service et le taux de déplacements devraient chacun plus ou moins doubler.

Le type de service offert en est un de "porte à porte sur demande" i.e. les personnes appellent pour être déplacées à une certaine heure (ou pour une certaine heure), d'un lieu d'origine à un autre lieu de destination. Du point de vue du transport, on distingue principalement trois types d'handicapés: ceux devant demeurer dans leur fauteuil roulant lors du déplacement et/ou possédant un fauteuil roulant non pliant (par exemple, un fauteuil motorisé), ceux pouvant utiliser un siège régulier lors du transport et possédant un fauteuil roulant pliant ainsi que ceux ayant besoin d'un soutien physique et/ou directionnel. Pour la plupart de ces personnes, le service "porte à porte" est la seule possibilité et ce même dans des conditions optimales d'accessibilité du service de transport en commun conventionnel, puisque les déplacements pour se rendre à l'arrêt et pour en repartir ne sont pas possibles, ou réalistes, pour eux.

La croissance fulgurante des services de transport adapté pose un défi con-

sidérable aux planificateurs de transport collectif. Ce projet de recherche concerne donc la réalisation d'un système de confection et d'optimisation d'itinéraires pour desservir la demande avec une flotte mixte de véhicules.

Dans les prochaines sections de ce chapitre, on retrouve l'énoncé du scénario d'opération utilisé, la formulation mathématique du problème, la revue de la littérature sur différentes variantes de problèmes de confection d'itinéraires, et finalement la méthodologie ainsi que la nouvelle approche proposée au problème.

1.2 Scénario d'opération

Le problème étudié est celui de la confection d'itinéraires et d'horaires de véhicules devant satisfaire un ensemble de requêtes de déplacement (clients). Chacune de ces requêtes exige l'embarquement à un lieu d'origine spécifique et le débarquement à un lieu de destination spécifique. Le scénario d'opération suppose que les requêtes sont connues à l'avance (par exemple, un jour avant celui du déplacement), qu'une flotte mixte de véhicules est disponible et que l'affectation des véhicules aux chauffeurs est connue. Le terme véhicule sera utilisé dans un sens très large puisqu'il est également associé à une pièce de travail d'un chauffeur. Les autres hypothèses ou considérations du scénario d'opération touchent les contraintes, les fonctions économiques ainsi que les données. Elles sont énumérées ci-dessous.

Les contraintes strictes—ne pouvant être violées—sont les suivantes:

1) Priorité : l'origine d'une requête est visitée avant sa destination.

2) Couplage : l'origine et la destination sont visitées par le même véhicule—donc pas de centre de correspondance.

3) Fenêtre horaire :

    a) les heures de service à l'origine et à la destination de chaque requête doivent respecter les fenêtres horaires associées.

b) les différentes périodes de disponibilité des véhicules—les pièces de travail des chauffeurs—doivent être respectées.

4) Service complet : autant que possible, toutes les requêtes doivent être desservies; certaines requêtes pourront être refusées seulement si le nombre de véhicules est insuffisant.

5) Disponibilité : seuls les véhicules disponibles pourront être considérés.

6) Capacité : la capacité de chacun des véhicules doit être respectée.

7) Dépôt : pour chaque véhicule, le dépôt de retour doit être celui du départ.

Les contraintes molles—pouvant être violées si nécessaire—sont les suivantes:

8) Temps maximum acceptable à bord d'un véhicule : l'intervalle de temps entre l'embarquement du client et son débarquement devrait être inférieur à une certaine borne fixée par la qualité de service permise. Ce temps maximum est limité de façon stricte—mais pas nécessairement de façon acceptable—par les fenêtres horaires.

9) Temps d'attente : il ne devrait pas y avoir d'attente d'un véhicule lorsqu'il y a des passagers à bord.

Les fonctions économiques globales—s'appliquant à l'ensemble des véhicules—peuvent inclure les termes suivants:

10) Coût de déplacement des véhicules.

11) Coût fixe d'utilisation des véhicules : nombre de véhicules utilisés.

12) Durée des itinéraires : temps entre le départ et le retour au dépôt.

13) Durée des périodes libres : une période libre est la période de temps où le véhicule est en attente. Les périodes libres durant lesquelles le véhicule a à son bord des passagers sont indésirables, tandis que les longues périodes libres avec aucun passager sont souhaitables et ce, en vue de l'insertion éventuelle de nouvelles requêtes après la confection des itinéraires.

Les fonctions économiques locales—s'appliquant à seulement un véhicule—peuvent inclure les termes suivants:

14) Qualité de service : délais et temps maximum acceptables à bord d'un véhicule.

15) Temps d'attente indésirable.

16) Coût de déplacement d'un véhicule.

Les principales données considérées sont les suivantes:

17) Les distances corrigées : elles sont calculées de façon à tenir compte des zones de congestion.

18) Temps de déplacement : ces temps sont calculés en fonction des distances, des périodes des déplacements et de la vitesse des véhicules. Ils pourraient inclure des temps fixes avant le départ et à l'arrivée aux lieux.

19) Arrêts : deux arrêts sont associés à chaque requête, un pour l'origine et un autre pour la destination.

20) Coûts de transition entre deux arrêts: ils dépendent de la fonction économique choisie.

21) Pièces de travail : elles sont caractérisées par une heure de début et une durée.

22) Fenêtres horaires des requêtes : elles sont calculées a priori à partir de l'heure de départ désirée ou de l'heure d'arrivée désirée qui est formulée dans la requête, du temps direct de déplacement ainsi que des politiques sur la qualité de service permise aux usagers.

23) Heures de service idéales : elles sont calculées à partir de l'heure désirée d'une requête et sont associées aux fenêtres horaires respectives.

24) Temps de service : ils correspondent aux temps utilisés pour embarquer ou débarquer les clients de chaque requête.

25) Temps maximum acceptables à bord d'un véhicule : ils sont associés à chaque requête. Ils sont calculés en fonction des temps directs de déplacement des requêtes et de la politique sur la qualité de service permise aux usagers.

26) Charges : elles sont associées aux requêtes. Elles sont multidimensionnelles puisqu'elles correspondent au nombre d'usagers par catégorie à déplacer (par exemple, les usagers avec ou sans fauteuil roulant).

27) Capacités : elles sont associées aux véhicules. Elles sont multidimensionnelles—pour accommoder différentes catégories d'usagers—et en plusieurs configurations, puisque les véhicules sont partiellement modifiables.

1.3. Formulation mathématique

Dans cette section, nous présentons la formulation simplifiée du problème de transport adapté, connu sous le nom de problème de cueillettes et livraisons avec fenêtres de temps (PDPTW : "Pick-up and Delivery Problem with Time Windows") ainsi que quelques importantes variantes dérivant du PDPTW. Pour simplifier la formulation, nous ne considérons qu'un seul dépôt, une flotte fixe de véhicules homogènes avec des capacités unidimensionnelles (il n'existe alors qu'une configuration possible), des temps de déplacement indépendants de la période de la journée et calculés a priori ainsi qu'un objectif de minimisation des coûts de déplacement.

Soient n requêtes indexées par i. On associe à l'origine de la requête i le noeud i et à la destination le noeud $n+i$. De plus, on associe le noeud 0 au dépôt origine et le noeud $2n+1$ au dépôt destination. Il peut arriver que deux noeuds de différentes requêtes correspondent au même lieu physique.

On pose $N = \{0, 1, 2, \ldots, 2n+1\}$ comme étant l'ensemble des noeuds de notre réseau et $P = P^+ \cup P^-$, $P^+ = \{1, 2, 3, \ldots, n\}$, $P^- = \{n+1, n+2, \ldots, 2n\}$ comme étant les ensembles des noeuds associés aux requêtes.

La requête i demande que d_i passagers (la charge) soient transportés du noeud i au noeud $n+i$. On dénote par $[a_i, b_i]$ la fenêtre de temps du noeud origine de la requête i et par $[a_{n+i}, b_{n+i}]$ la fenêtre de temps du noeud destination de la même requête. Les temps de service s_i et s_{n+i} sont associés respectivement au noeud i et au noeud $n+i$. Finalement, r_i représente le déplacement maximal acceptable pour la requête i.

On définit par $V = \{1, 2, \ldots, m\}$ l'ensemble des véhicules à utiliser, par D la capacité de ces véhicules et par K le coût fixe encouru par l'utilisation d'un véhicule. On dénote par $[a_0, b_0]$ la fenêtre de temps pour le départ des véhicules du dépôt origine et par $[a_{2n+i}, b_{2n+i}]$ la fenêtre de temps pour le retour des véhicules au dépôt destination.

On représente par t_{uw} et c_{uw}, $u \neq w$ et $u, w \in N$ le temps de déplacement et le coût de déplacement du noeud u au noeud w, respectivement. L'arc (u, w) est présent dans le réseau s'il satisfait:

(i) l'exigence minimale en temps:

$$a_u + s_u + t_{uw} \leq b_w \quad u, w \in P \tag{1}$$

ii) l'exigence minimale en capacité:

$$d_u + d_w \leq D \quad \text{si} \quad u \neq w \quad \text{et} \quad u, w \in P^+ \quad \text{ou} \tag{2}$$

$$d_{u-n} + d_{w-n} \leq D \quad \text{si} \quad u \neq w \quad \text{et} \quad u, w \in P^- \quad \text{ou} \tag{3}$$

$$d_u + d_{w-n} \leq D \quad \text{si} \quad u \in P^+, w \in P^-. \tag{4}$$

Trois types de variables sont utilisées dans la formulation mathématique: les variables binaires de flot X^v_{uw}, $u \neq w$, $u, w \in N$ et $v \in V$; les variables de temps T_u, T^v_0, T^v_{2n+1}, $u \in P$ et $v \in V$ et les variables de charge Y_u, $u \in P$. La variable binaire X^v_{uw} prend la valeur 1 si le véhicule v utilise l'arc (u, w) et prend la valeur 0 sinon. La variable T_u prend la valeur de l'heure de service au noeud u. Tandis que T^v_0 et T^v_{2n+1} prendront respectivement les valeurs des heures où le véhicule part du dépôt et arrive au dépôt. La variable Y_u représente

la charge totale (le nombre de passagers) dans le véhicule immédiatement après avoir quitté le noeud u.

Formulation mathématique du problème:

$$\min \sum_{v\in V}\sum_{u\in N}\sum_{w\in N} c_{uw}\cdot X^v_{uw} \tag{5}$$

sujet à

$$\sum_{v\in V}\sum_{w\in N} X^v_{uw} = 1 \qquad u\in P^+ \tag{6}$$

$$\sum_{w\in N} X^v_{uw} - \sum_{w\in N} X^v_{uw} = 0 \qquad u\in P,\ v\in V \tag{7}$$

$$\sum_{w\in P^+} X^v_{0w} = 1 \qquad v\in V \tag{8}$$

$$\sum_{u\in P^-} X^v_{u,2n+1} = 1 \qquad v\in V \tag{9}$$

$$\sum_{w\in N} X^v_{uw} - \sum_{w\in N} X^v_{w,n+u} = 0 \qquad u\in P,\ v\in V \tag{10}$$

$$t_{u,n+u} \le T_{n+u} - T_u - s_u \le r_u \qquad u\in P^+ \tag{11}$$

$$X^v_{uw} = 1 \Rightarrow T_u + s_u + t_{uw} \le T_w \qquad u,\ w\in P,\ v\in V \tag{12}$$

$$X^v_{0w} = 1 \Rightarrow T^v_0 + t_{0w} \le T_w \qquad w\in P^+,\ v\in V \tag{13}$$

$$X^v_{u,2n+1} = 1 \Rightarrow T_u + s_u + t_{u,2n+1} \le T^v_{2n+1} \qquad u\in P^-,\ v\in V \tag{14}$$

$$a_u \le T_u \le b_u \qquad u\in P \tag{15}$$

$$a_0 \le T^v_0 \le b_0 \qquad v\in V \tag{16}$$

$$a_{2n+1} \le T^v_{2n+1} \le b_{2n+1} \qquad v\in V \tag{17}$$

$$X^v_{uw} = 1 \Rightarrow Y_u + d_w = Y_w \qquad u\in P,\ w\in P^+,\ v\in V \tag{18}$$

$$X^v_{uw} = 1 \Rightarrow Y_u - d_{w-n} = Y_w \qquad u\in P,\ w\in P^-,\ v\in V \tag{19}$$

$$X^v_{0w} = 1 \Rightarrow d_w = Y_w \qquad w\in P^+,\ v\in V \tag{20}$$

$$0 \le Y_u \le D \qquad u\in P^+ \tag{21}$$

X^v_{uw} binaire $\qquad u,\ w\in N,\ v\in V$ (22)

L'objectif du problème consiste à minimiser la somme des coûts de déplacement. Les contraintes (6)–(9) et (22) correspondent aux contraintes d'un problème de flot à coût minimum avec plusieurs commodités. La contrainte de couplage (10) assure que le même véhicule v visite les noeuds u et $n+u$. Les contraintes (11) incluent par l'inégalité de gauche la contrainte de priorié qui force le noeud u à être visité avant le noeud $n+u$ et par l'inégalité de droite, la contrainte de déplacement maximum pour la requête u. Les contraintes (12)–(14) décrivent les exigences de comptabilité entre les itinéraires et l'horaire tandis que les contraintes (15)–(17) établissent les contraintes d'horaire. De même, les contraintes (18)–(20) décrivent les exigences de comptabilité entre les itinéraires et la charge des véhicules tandis que les contraintes (21) établissent les contraintes de capacité.

Notons que la formulation n'inclut pas de contraintes explicites d'élimination de sous-tours puisque les contraintes (12) assurent l'élimination de sous-tours (Desrosiers et al. 1984). Les contraintes (12) peuvent être réécrites dans une forme linéaire équivalente en utilisant une grande valeur M:

$$T_u + s_u + t_{uw} - T_w \leq (1 - X^v_{uw}) \cdot M \qquad u, w \in P,\ v \in V \quad (23)$$

Les contraintes d'élimination de sous-tours de Miller, Tucker et Zemlin (1960) pour le voyageur de commerce représentent un cas spécial des contraintes (23) lorsque $t_{uw} = 1, \quad s_u = 0$ et $M = |P|$.

Examinons maintenant quelques variantes importantes qui dérivent de la formulation présentée. Cette formulation est également connue sous l'appellation anglaise "Multi-vehicle many-to-many dial-a-ride problem with time windows". Pour considérer la minimisation du nombre de véhicules on doit ajouter à l'objectif le terme $\sum_{v \in V} \sum_{w \in P^+} K \cdot X^v_{0w}$ et on remplace le signe d'égalité par le signe de plus petit ou égal dans les contraintes d'utilisation des véhicules (8) et (9).

Si nous posons $m = 1$ et éliminons les contraintes de couplage (10), nous obtenons la variante pour un seul véhicule: "the single vehicle many-to-many dial-a-ride problem with time windows".

Pour la suite des variantes, définissons $P = P^+ = P^- = \{1, 2, \ldots, n\}$ et $N = P \cup \{0, n+1\}$. Pour obtenir le problème du transport de marchandises avec fenêtres horaires (VRPTW: "Vehicle Routing Problem with Time Windows"), on élimine les contraintes de couplage (10), les contraintes de priorité et de déplacement maximum (11) ainsi que les contraintes (19) définissant une partie des contraintes de compatibilité entre les charges et les itinéraires. Pour les problèmes de cueillette, on aura $d_w \geq 0$ seulement. Pour ceux de livraison, on aura $d_w \leq 0$ avec les contraintes (20) remplacées par

$$X^v_{0w} = 1 \Rightarrow D + d_w = Y_w \qquad w \in P^+,\ v \in V. \quad (20')$$

En éliminant les contraintes de couplage (10), les contraintes de priorité et de déplacement maximum (11) ainsi que les contraintes de capacité (18)–(21), on obtient le problème du voyageur de commerce multiple avec fenêtres horaires (m-TSPTW : "Multiple Vehicle Traveling Salesman Problem with Time Windows"). En posant de plus $m = 1$, on obtient le problème du voyageur de commerce avec fenêtres horaires (TSPTW : "Traveling Salesman Problem with Time Windows"). Finalement, de la formulation présentée, on obtient le plus court chemin avec fenêtres horaires (SPPTW: "Shortest Path Problem with Time Windows") présenté dans Desrosiers et al. (1983), en utilisant les relations (5), (7)–(9), (12)–(17), (22) et en définissant $m = 1$.

1.4. Revue de la littérature

Nous présentons dans les cinq sections suivantes la revue de la littérature pour les variantes introduites à la section précédente. L'ensemble des articles suivants offrent une revue de la littérature plus complète sur les problèmes de confection d'itinéraires en général: Magnanti (1981), Bodin et al. (1983), Laporte et Nobert (1987), Desrochers et al. (1988), et Solomon et Desrosiers (1988).

1.4.1 Le problème du voyageur de commerce avec fenêtres horaires (TSPTW)

Le TSPTW est un VRPTW avec seulement un véhicule, sans capacité. Christofides et al. (1981b) et Baker (1983) présentent des algorithmes optimaux tandis que Savelsbergh (1986) présente un algorithme heuristique basé sur le concept de k-échanges. Ce dernier ayant prouvé que même le problème de trouver une solution réalisable au TSPTW est un problème NP-complet.

Les travaux de Christofides et al. (1981b) explorent la relaxation d'états pour le TSPTW avec des approches de programmation dynamique. Les bornes inférieures obtenues des récursions relaxées pourraient être améliorées avec l'utilisation de méthodes d'optimisation du sous-gradient et de la dimension maximale des états ("State-Space Ascent"). On pourrait les utiliser dans un algorithme de séparation et d'évaluation progressive ("Branch and bound") pour résoudre le TSPTW.

Les travaux de Baker (1983) présentent un algorithme de séparation et d'évaluation progressive pour une nouvelle formulation du TSPTW exprimée en termes de variables de temps. L'algorithme exploite la structure du dual de la relaxation du model proposé, lequel est un problème de plus long chemin sur un réseau acyclique. Celui-ci fournit une borne inférieure au problème original. Le branchement à un noeud de l'arbre d'énumération crée deux sous-problèmes correspondant au cas où l'arrivée à la requête i précède l'arrivée à la requête j et vice versa.

1.4.2 Le problème du voyageur de commerce multiple avec fenêtres horaires (m-TSPTW)

Le m-TSPTW est un VRPTW avec m véhicules, sans capacité. Lorsque les fenêtres de temps sont fermées, i.e., $a_u = b_u$, $u \in P$, alors les problèmes pour minimiser le nombre de véhicules ou la distance, sont facilement résolubles (par le Théorème de décomposition de Dilworth et par l'utilisation d'une formulation de flot dans un réseau, respectivement).

Plusieurs travaux de recherche sont parus dans le contexte de la minimisation du nombre de véhicules dans les domaines du transport aérien et scolaire. Gertsbakh et Stern (1978) explorent un cas spécial du m-TSPTW et présentent une méthode exacte basée sur une formulation de programmation linéaire en nombres entiers, ainsi qu'un algorithme heuristique. Levin (1971) présente, dans le contexte du transport aérien, une formulation de programmation linéaire en nombres entiers avec des fenêtres de temps discrétisées. La méthode est vue comme une heuristique pour les problèmes avec fenêtres de temps continues. Les solutions sont obtenues en résolvant la relaxation linéaire du problème, laquelle produit souvent des solutions entières. Dans le même contexte, Martin (1981) présente un algorithme de type énumération vers l'avant, avec des règles d'exclusion pour générer un horaire réalisable, lorsque le nombre d'avions est fixé. Swersey et Ballard (1983) ont utilisé la formulation et l'approche de Levin (1971) pour résoudre avec succès des problèmes de transport scolaire. Par ajustement de quelques valeurs fractionnaires, ils ont obtenu des solutions entières sans augmentation du nombre d'autobus nécessaires.

Avec une fonction économique plus générale, incluant la minimisation des coûts fixes et la minimisation des temps de déplacement, le problème de transport scolaire est équivalent au m-TSPTW. Orloff (1976) présente un algorithme heuristique de deux phases pour le transport scolaire. Dans la première phase, un horaire réalisable est créé. Une suite de problèmes de couplage ("Matching") est résolue pour obtenir des segments de plus en plus longs. Dans la phase finale, un algorithme de type 3-opt est utilisé pour améliorer l'horaire réalisable. Bodin et Berman (1979) ainsi que Desrosiers et al. (1986b) proposent aussi des algorithmes

heuristiques. Les premiers partitionnent l'horizon du temps et résolvent une suite de problèmes de transport pour lier les itinéraires dans les périodes adjacentes tandis que chez les seconds, l'horizon du temps n'est pas partitionné a priori.

Dans le contexte du transport urbain, Soumis et al. (1985) proposent une méthode exacte dans laquelle les contraintes d'horaire sont relaxées. Le problème de réseau résultant est résolu et un branchement de type Carpaneto et Toth (1980) est utilisé sur les variables de flot pour satisfaire les contraintes d'horaire. Des problèmes comprenant jusqu'à 158 requêtes ont été résolus facilement. Une stratégie de branchement plus efficace est proposée dans Desrosiers et al. (1985) pour des fenêtres de temps plus larges. La stratégie est basée sur la séparation des fenêtres de temps.

Finalement, Desrosiers et al. (1984) proposent un algorithme primal qui utilise l'approche de génération de colonnes. Le problème maître est un problème de partitionnement où les colonnes représentent les itinéraires admissibles. Le sous-problème générant les colonnes est un plus court chemin avec fenêtres horaires (SPPTW). L'expérience des auteurs indique que la relaxation linéaire du problème de partitionnement donne une très bonne borne inférieure et ce, même lorsque le nombre de noeuds augmente ou que les largeurs des fenêtres de temps augmentent. La borne inférieure est améliorée à l'aide d'une coupe sur le nombre de véhicules qui exigent l'utilisation d'un nombre entier de véhicules. Dans tous les cas, le nombre optimal de véhicules a été obtenu. Des problèmes comprenant jusqu'à 151 requêtes ont été résolus. Le SPPTW survient également dans une approche duale par relaxation lagrangienne (Desrosiers et al. 1988b) où les contraintes exigeant la visite de chaque noeud sont introduites dans l'objectif. Cette approche requiert moins d'espace mémoire que l'approche primale et les résultats numériques sont équivalents.

1.4.3 Le problème du plus court chemin avec fenêtres horaires (SPPTW)

Ce problème apparaît naturellement comme sous-problème pour les approches de génération de colonnes basées sur une formulation de partitionnement d'ensemble ou pour les approches de relaxation lagrangienne pour résoudre le m-TSPTW. Le SPPTW a été étudié intensivement par Desrosiers et al. (1983) et Desrochers et Soumis (1988a, 1988b).

Desrosiers et al. (1983) ont développé une généralisation de l'algorithme de programmation dynamique de Ford-Bellman-Moore pour le plus court chemin standard, tandis que Desrochers et Soumis (1988a) ont développé une généralisation de l'algorithme de Dijkstra qui est plus rapide. Desrochers et Soumis (1988b) ont aussi présenté un algorithme primal-dual de réoptimisation lequel est très intéressant dans le contexte de génération de colonnes multiples disjointes, pour le m-TSPTW. Finalement Desrochers (1988) généralise l'approche primale-duale au plus court chemin avec contrainte de ressources multiples.

Un autre plus court chemin contraint est celui apparaissant comme sous-problème dans l'approche de génération de colonnes pour résoudre le problème avec cueillettes et livraisons. Dumas et Desrosiers (1988) présentent un algorithme de programmation dynamique efficace lequel est utilisé pour résoudre des problèmes comportant jusqu'à 55 requêtes (110 noeuds).

Dans la littérature, la programmation dynamique a été utilisée avec beaucoup de succès pour obtenir des solutions optimales entières au problème de plus court chemin avec contraintes. Lorsque les contraintes sont serrées, les problèmes de plus court chemin sont facilement résolus par programmation dynamique puisque l'ensemble des états réalisables est relativement petit. De plus, en résolvant le sous-problème en nombre entier, on réduit l'écart d'intégralité ("Duality gap") du problème original et on produit de meilleures bornes. Le gain sur la qualité de la borne est particulièrement remarquable pour les problèmes très contraints.

1.4.4 Le problème du transport de marchandise avec fenêtres horaires (VRPTW)

Les premiers travaux parus sur le sujet ont présenté des méthodes heuristiques développées dans des contextes d'applications. On retrouve Pullen et Webb (1967) (horaire de service des chauffeurs de camions), Knight et Hofer (1968) (compagnie de transport contractante) et Madsen (1976) (compagnie de distribution de journaux et de magazines).

Les seules méthodes exactes ont été proposés par Kolen et al. (1987) et Jorhsten et al. (1986). Ces méthodes ne permettent de résoudre que de très petits problèmes. Les difficultés inhérentes au problème ont conduit aux développements de plusieurs heuristiques.

Solomon (1986, 1987) a élaboré et analysé une variété d'heuristiques de construction d'itinéraires pour le VRPTW. Les résultats présentés dans Solomon (1987) indiquent que les algorithmes d'insertion séquentielle dans le temps et l'espace (généralisation de l'approche de Mole et Jameson (1976) au VRP), fournissent de très bons résultats dans un grand nombre d'applications pour le VRPTW. L'excellente performance de telles heuristiques peut être expliquée par le fait que l'affectation des clients aux véhicules est l'aspect le plus important pour les problèmes de routage (confirmé par le succès de l'heuristique d'affectation généralisée de Fisher et Jaikumar (1981)) – alors que l'ordonnancement est l'aspect le plus important pour les problèmes de routage avec des contraintes restrictives d'horaire.

Les travaux de Baker et Schaffer (1986), Cook et Russel (1978), Solomon et al. (1986) et Savelsbergh (1986) présentent des heuristiques d'amélioration utilisant des procédures d'échanges. On retrouve entre autre des généralisations du 2-opt, 3-opt et k-opt de Lin (1965) et du Or-opt de Or (1976).

Finalement, dans les travaux sur des méthodes exactes, Kolen et al. (1987) généralisent l'approche de q-chemin de Christofides et al. (1981a) au problème avec fenêtres horaires. Le plus gros problème résolu est de 14 requêtes et 4 véhicules avec de petites fenêtres horaires. Egalement, Jornsten et al. (1986) proposent une

relaxation lagrangienne pour le calcul d'une borne inférieure pour le VRPTW. Utilisant l'approche de "variable splitting", cette relaxation nécessite des solutions entières pour deux types de sous-problèmes: le SPPTW et le problème d'affectation généralisée. Aucun résultat numérique n'a été présenté.

1.4.5 Le problème de cueillettes et livraisons avec fenêtres horaires (PDPTW)

Le PDPTW est une généralisation du VRPTW. Le VRPTW est un problème de cueillettes ou de livraisons tandis que le PDPTW est un problème de cueillettes et de livraisons. Le PDPTW implique des contraintes supplémentaires de priorité et de couplage. C'est le problème décrit par les relations (5)–(22).

A notre connaissance, le seul algorithme exact actuellement proposé est celui de Dumas (1985). Cet algorithme utilise le concept de génération de colonnes avec des sous-problèmes de plus court chemin avec contraintes de capacité, de priorité, de couplage et de fenêtres horaires (Dumas et Desrosiers 1988). L'algorithme peut considérer une flotte mixte de véhicules avec plusieurs dépôts. Des problèmes jusqu'à 55 requêtes (110 noeuds) et 25 véhicules avec contraintes de capacité restrictives et des fenêtres horaires peu restrectives ont été résolus.

Plusieurs travaux sont parus dans le cas d'un seul véhicule. Psaraftis (1983a) présente un algorithme exact de programmation dynamique. La fonction économique utilisée est celle de la minimisation de la durée de l'itinéraire. La complexité de l'algorithme est de $0(n^2 3^n)$, ce qui limite la taille des problèmes résolubles à environ 10 requêtes. Psaraftis (1983b, 1983c) a aussi développé plusieurs heuristiques de complexité polynomiale pour traiter des tailles de problèmes plus réalistes. Sexton et Bodin (1985a, 1985b) présentent une formulation de programmation non linéaire avec variables mixtes (entières et continues). Une décomposition de Benders est utilisée pour partitionner le problème en deux. On obtient alors un problème de confection d'itinéraires ("routing problem") et un problème de confection d'horaire ("scheduling problem"). Le premier est un problème non linéaire en nombre entier et le second est un problème de réseau. La

procédure entière est heuristique due à la difficulté de résoudre le problème de confection d'itinéraires. Des problèmes de 7 à 20 requêtes ont été résolus. Sexton et Choi (1986) présentent sensiblement la même chose que Sexton et Bodin (1985a) mais dans un contexte de transport de marchandises avec fenêtres horaires molles (qui peuvent être violées) où l'algorithme de confection d'horaire a été généralisé.

Pour la version avec plusieurs véhicules, nous présentons quatre études. Le travail de Roy et al. (1984a, 1984b) est basé sur une procédure d'insertion en parallèle. Les requêtes sont ordonnées dans le temps puis traitées grossièrement dans cet ordre. Un horizon fuyant est utilisé pour limiter le choix des requêtes. Les itinéraires des véhicules sont initialisés ou prolongés au fur et à mesure que l'horizon fuyant avance. Jaw et al. (1982) présentent une heuristique de type répartition des requêtes entre les véhicules en premier et puis ordonnancement des arrêts d'un véhicule en second ("clusters first - routing second"). Au lieu de fonctionner avec un horizon fuyant, l'horizon est partitionné a priori. Des contraintes d'horaire molles sont considérées. Jaw et al. (1986) ont également utilisé une procédure d'insertion en parallèle. La procédure proposée semble très efficace; des problèmes jusqu'à 2 600 requêtes ont été résolus. Finalement, Bodin et Sexton (1986) présentent aussi une procédure de type répartition en premier et ordonnancement en second. Les requêtes sont d'abord affectées aux véhicules puis l'algorithme d'un véhicule basé sur la décomposition de Benders (Sexton et Bodin 1985a, 1985b) est appliqué à chacun des véhicules.

1.5. Méthodologie et approche proposées

Le problème étudié est celui de la confection d'itinéraires et d'horaires de véhicules dans le contexte du transport adapté (DARP:"Dial-A-Ride Problem"). C'est un contexte particulier d'application du PDPTW. Cette thèse présente une approche de résolution pouvant traiter des problèmes de grande taille (par exemple, 2000 requêtes et 150 véhicules) et pouvant fournir des solutions de qualité supérieure à celles fournies par les approches déjà existantes.

La revue de la littérature ne nous suggère, évidemmment, aucune approche optimale pour la taille des problèmes que nous désirons résoudre. La plupart des approches proposées décomposent le problème global pour obtenir des problèmes restreints plus faciles à résoudre. Ainsi, le problème se décompose naturellement en trois problèmes restreints, qui ne sont pas indépendants. Ces problèmes restreints sont:

i) la répartition des requêtes entre les véhicules ("Clustering")

ii) l'ordonnancement des arrêts de chaque véhicule ("Routing")

iii) la confection de l'horaire de chaque véhicule ("Scheduling").

Le problème de répartition des requêtes est le plus important et en même temps le plus difficile, puisqu'il doit assurer l'existence d'un ordonnancement réalisable pour chaque véhicule. Bodin et Sexton (1986) et Jaw et al. (1982) ont proposé des algorithmes pour répartir les requêtes. Ces méthodes ne peuvent toutefois pas assurer l'existence d'un ordonnancement réalisable pour chaque véhicule dans le contexte de fenêtres horaires strictes. Ceci est dû au fait qu'aux trois niveaux, les décisions sont prises de façon indépendante.

Les principales méthodes heuristiques proposées pour résoudre le problème de répartition sont basées sur l'insertion. Ces méthodes résolvent à la fois le premier et le deuxième problème en construisant des itinéraires réalisables pour chaque véhicule. L'insertion se fait soit en séquentielle (Wilson et al. 1971, 1976, 1977 et Hung et al. 1982) soit en parallèle (Roy et al. 1984a, 1984b et Jaw et al. 1986). L'insertion séquentielle a tendance à construire des itinéraires productifs

au début et des itinéraires de moins bonne qualité par la suite. Quant à l'insertion parallèle, elle construit des itinéraires beaucoup plus équilibrés. Il est cependant très difficile d'obtenir une bonne répartition des requêtes par des méthodes d'insertion; ces méthodes ont une vue trop locale du problème. Des procédure d'échanges ont été appliquées pour améliorer la répartition.

Le problème d'ordonnancement est également difficile mais comme on traite un véhicule à la fois, la taille des problèmes à résoudre est beaucoup moindre que celle du problème de répartition. Son implication sur la solution est locale puisqu'elle influence l'itinéraire d'un seul véhicule à la fois. Des approches exactes ont été proposées (Psaraftis 1980 et 1983a) pour résoudre ce problème; cependant seuls des problèmes de petite taille—8 à 10 requêtes—ont pu être résolus. Des méthodes ont également été proposées pour résoudre les deux derniers problèmes à l'aide de la technique de décomposition de Benders (Sexton et Bodin 1985a, 1985b et Sexton et Choi 1986). La difficulté pour optimiser le sous-problème de confection d'itinéraires a forcé les auteurs à résoudre le problème de façon heuristique.

Le problème de confection de l'horaire d'un véhicule étant donné son itinéraire est facile mais non trivial. Une approche a été proposée dans le contexte du transport adapté où les inconvénients sur les heures de service sont modélisées par des fonctions linéaires (Sexton et Bodin 1985a). Les auteurs ont montré que le dual de leur modèle se formule comme un problème de flot et qu'il peut être résolu en temps linéaire.

L'approche proposée se divise en trois phases, correspondant aux problèmes restreints précédents. La première répartit les requêtes entre les véhicules (**phase de répartition** ou "clustering"). La deuxième établit les itinéraires de chacun des véhicules (**phase d'ordonnancement** ou "routing") et la troisième détermine l'horaire de chacun des itinéraires (**phase de confection d'horaire** ou "scheduling"). Par contre, les trois niveaux de décisions ne sont pas pris de façon disjointe. Pour assurer la faisabilité des phases aux niveaux inférieurs, il doit y avoir intéraction entre les différents types de décisions.

La première phase, effectuant la répartition des requêtes entre les véhi-

cules, se divise en 2 étapes. Dans la première étape intervient l'important concept de tâche. Une procédure prend des décisions locales sur le regroupement et l'ordonnancement des requêtes de façon à regrouper dans des portions d'itinéraires faisables des requêtes voisines (de 3 à 4 requêtes en moyenne). Chacune des portions d'itinéraire forme ainsi une tâche. Les propriétés recherchées de ces tâches sont: (1) la faisabilité: la portion d'itinéraire de chaque tâche doit satisfaire l'ensemble des requêtes de la tâche en respectant les contraintes de fenêtres de temps, de couplage, de priorité et de capacités du véhicule, (2) l'efficacité: la somme des temps de déplacement des véhicules à l'intérieur des tâches est assez proche du minimum. Par exemple, si $+i$ représente l'embarquement du client de la requête i et $-i$ son débarquement alors les portions d'itinéraires suivantes sont des tâches: $+1 \rightarrow -1$, $+1 \rightarrow +2 \rightarrow -2 \rightarrow -1$ et $+1 \rightarrow +2 \rightarrow -2 \rightarrow +3 \rightarrow -1 \rightarrow -3$ mais pas ceux-ci: $+1 \rightarrow +3 \rightarrow -1$ et $-1 \rightarrow +1$ puisque les contraintes de priorité et de couplage ne sont pas respectées. Plusieurs procédures heuristiques peuvent être utilisées pour créer un ensemble de tâches satisfaisant les propriétés ci-haut mentionnées. Par exemple, les requêtes pourraient être tout simplement regroupées selon des critères de proximité et de faisabilité. Nous avons plutôt utilisé une procédure qui effectue le morcellement des itinéraires d'une solution initiale aux endroits où le véhicule se déplace sans passagers. Il est facile de se convaincre que les tâches ainsi créées ont la propriété de faisabilité. Cette solution initiale peut aussi bien être une solution manuelle, une solution générée par un des deux algorithmes déjà existant ou, après une itération, la solution produite par notre approche (ré-optimisation).

Dans la deuxième étape de la deuxième phase, on répartit et ordonne les tâches entre les véhicules en solutionnant un problème de voyageur de commerce multiple avec fenêtres de temps (m-TSPTW) et contraintes de ressources. A ce niveau les décisions globales sur la répartition sont effectuées en prenant en considération les coûts de répartition, les coûts d'ordonnancement ainsi que la faisabilité des phases suivantes. Desrosiers et al. (1984) ont proposé une approche optimale pour le m-TSPTW avec un seul type de véhicules; cette approche peut traiter rapidement des problèmes d'une centaine de tâches. L'approche est généralisée au chapitre 2 pour considérer une flotte mixte de véhicules avec

plusieurs dépots ainsi que des contraintes de compatibilité entre les véhicules et les tâches. Le nombre estimé de tâches étant de l'ordre de 500 à 1000 pour des problèmes de 2000 requêtes, il est impossible de penser résoudre le m-TSPTW directement. Ainsi, nous présentons au chapitre 5 une décomposition géographique et temporelle du problème qui permet d'obtenir une suite de m-TSPTW de taille raisonnable à résoudre. Cette approche génère des itinéraires sur les tâches et assure l'existence d'itinéraires et d'horaires faisables (sur les requêtes) pour chaque véhicule. Toutefois, un itinéraire efficace sur les tâches n'implique pas un itinéraire efficace sur les requêtes.

Dans la deuxième phase, on réordonne optimalement les requêtes de chaque véhicule en solutionnant un problème de voyageur de commerce avec contraintes additionnelles de temps, de priorité et de capacité (1-PDPTW). Les algorithmes exacts proposés jusqu'à présent n'ont permis de résoudre que de petits problèmes—moins de 15 requêtes. Aucun n'a exploité la contrainte exigeant que les requêtes affectées à un véhicule doivent être toutes desservies par celui-ci. Cette contrainte est très restrictive et elle permet d'élaborer plusieurs critères efficaces d'élimination d'états (dans le contexte de la programmation dynamique). Au chapitre 3, nous proposons un algorithme exact de programmation dynamique, qui permet de résoudre plus efficacement des problèmes de grande taille—plus de 40 requêtes.

Les itinéraires construits à partir du regroupement des requêtes en tâches, de l'affectation des tâches aux véhicules et de la réoptimisation de l'ordonnancement des arrêts sont faisables. La procédure est complétée dans une troisième phase par l'utilisation d'un algorithme exact de confection d'horaire déterminant les heures de services aux arrêts de l'itinéraire. Il y a généralement plusieurs horaires qui sont compatibles avec l'itinéraire et les fenêtres horaires. Le temps d'attente du véhicule peut être réparti de plusieurs façons le long de cet itinéraire. Le problème à résoudre est celui de choisir l'horaire qui minimise les inconvénients associés aux fenêtres horaires à chacun des noeuds. Les inconvénients sont modélisés par des fonctions convexes sur les heures de service. Les fonctions convexes permettent de générer des solutions moins extrémistes que les fonctions linéaires considérées par Sexton et al. (1985a, 1986). L'algorithme dual proposé est présenté

au chapitre 4.

Afin de réaliser les tests numériques, nous avons dû développer une fonction d'estimation des temps de déplacement fournissant des temps aux 5 minutes comparables à ceux évalués par les répartiteurs. L'annexe A présente des modèles statistiques d'estimation où la variable dépendante (le temps de déplacement) est une variable discrète. Un algorithme original permet de résoudre ce problème mathématique.

Finalement, l'approche complète est testée sur des données disponibles provenant d'une journée réelle d'opération à la Wheel-Trans Service de Toronto. Le problème contient 1890 requêtes ainsi qu'un ensemble de 133 pièces de travail. Une nouvelle fonction économique est également présentée pour favoriser les solutions permettant une insertion de requêtes ultérieures.

1.6 Références

E. BAKER (1983) An Exact Algorithm for the Time Constrained Traveling Salesman Problem. *Operations Research* 31, 938–945.

E. BAKER and J. SCHAFFER (1986) Computational Experience with Branch Exchange Heuristics for Vehicle Routing Problems with Time Window Constraints. *American Journal of Mathematical and Management Sciences* 6, 261–300.

BELISLE (1986) Le transport des personnes à mobilité réduite en milieu urbain. Document de présentation de projet de recherche, Ecole polytechnique de Montréal.

L. BODIN and L. BERMAN (1979) Routing and Scheduling of School Buses by Computer. *Transportation Science* 13, 113–129.

L. BODIN, B. GOLDEN, A. ASSAD and M. BALL (1983) Routing and Scheduling of Vehicles and Crews: The State of the Art. *Computers and Operations Research* 10, 62–212.

L. BODIN and T. SEXTON (1986) The Multi-Vehicle Subscriber Dial-a-Ride Problem. *TIMS Studies in the Management Sciences* 22, 73–86.

G. CARPANETO and P. TOTH (1980) Some New Branching and Bounding Criteria for the Asymmetric Traveling Salesman Problem. *Management Science* 26, 736–743.

N. CHRISTOFIDES, A. MINGOZZI and P. TOTH (1981a) Exact Algorithms for the Vehicle Routing Problem, Based on Spanning Tree and Shortest Path Relaxations. *Mathematical Programming* 20, 255–282.

N. CHRISTOFIDES, A. MINGOZZI and P. TOTH (1981b) State Space Relaxation Procedures for the Computation of Bounds to Routing Problems. *Networks* 11, 145–164.

G. CLOUTIER et D. MONGRAIN (1986) Le transport: un service sans dis-

crimination ni privilège au Québec. Présentation, quatrième conférence internationale sur la mobilité et le transport des personnes agées ou handicapées, Vancouver.

T. COOK and R. RUSSEL (1978) A Simulation and Statistical Analysis of Stochastic Vehicle Routing with Timing Constraints. *Decision Sciences* 9, 673–687.

M. DESROCHERS (1988) An Algorithm for the Shortest path Problem with Resource Constraints. École de Hautes Étude Commerciales de Montréal, Cahier du GERAD G-88-27, 32 pages.

M. DESROCHERS and F. SOUMIS (1988a) A Generalized Permanent Labeling Algorithm for the Shortest Path Problem with Time Windows. *INFOR* 26, no.3, 191–212.

M. DESROCHERS and F. SOUMIS (1988b) A Reoptimization Algorithm for the Shortest Path Problem with Time Windows. *European Journal of Operations Research* 35, 242–254.

M. DESROCHERS, J.K. LENSTRA, M.W.P. SAVELSBERGH and F. SOUMIS (1988) Vehicle Routing with Time Windows: Optimization and Approximation, *TIMS Studies in Management Science* 16, 65–84.

J. DESROSIERS, P. PELLETIER et F. SOUMIS (1983) Plus Court Chemin avec Contraintes d'Horaires. *RAIRO* 17, 357–377.

J. DESROSIERS, F. SOUMIS and M. DESROCHERS (1984) Routing With Time Windows by Column Generation. *Networks* 14, 545–565.

J. DESROSIERS, F. SOUMIS, M. DESROCHERS and M. SAUVÉ (1985) Routing and Scheduling with Time Windows Solved by Network Relaxation and Branch-and-Bound on Time Variables. In Computer Scheduling of Public Transport 2, J.M. Rousseau (ed.), North-Holland (1985), 451–469.

J. DESROSIERS, Y. DUMAS and F. SOUMIS (1986a) A Dynamic Programming Solution of the Large-Scale Single-Vehicle Dial-a-Ride Problem with

Time Windows. *American Journal of Mathematical and Management Sciences* 6, 301–325.

J. DESROSIERS, J. FERLAND, J. ROUSSEAU, G. LAPALME et L. CHAPLEAU (1986b) TRANSCOL: A Multi-Period School Bus Routing and Scheduling System. *TIMS Studies in the Management Sciences* 22, 47–71.

J. DESROSIERS, Y. DUMAS and F. SOUMIS (1988a) The Multiple Vehicle DIAL-A-RIDE Problem, *Lecture Notes in Economics and Mathematical System* 308: Computer-Aided Transit Scheduling, J.R. Daduna, A. Wren (Editor), Springer-Verlag, 15–27.

J. DESROSIERS, F. SOUMIS and M. SAUVE (1988b) Lagrangian Relaxation Methods for Solving the Minimum Fleet Size Multiple Traveling Salesman Problem with Time Windows. Forthcoming in *Management Science*.

Y. DUMAS (1985) Confection d'itinéraires de véhicules en vue du transport de plusieurs origines à plusieurs destinations. Mémoire de maitrise, Publication #434, Centre de Recherche sur les Transports, Université de Montréal.

Y. DUMAS and J. DESROSIERS (1988) A Shortest Path Problem for Vehicle Routing with Pick-up, Delivery and Time Windows. *Lecture Notes in Economics and Mathematical System* 302: Computer-Aided Transit Scheduling, J.R. Daduna, A. Wren (Editor), Springer-Verlag, 144–157.

M. FISHER and R. JAIKUMAR (1981) A Generalized Assignment Heuristic for Vehicle Routing. *Networks* 11, 109–124.

M. FISHER, A. GREENFIELD, R. JAIKUMAR and P. KEDIA (1982) Real-Time Scheduling of a Bulk Delivery Fleet: Practical Application of Lagrangian Relaxation. Working Paper 82-10-11, Department of Decision Sciences, University of Pennsylvania.

I. GERTSBAKH and H. STERN (1978) Minimal Resources for Fixed and Variable Job Schedules. *Operations Research* 26, 68–85.

H. HUNG, R. CHAPMAN, W. HALL and E. NEIGUT (1982) A Heuristic

Algorithm for Routing and Scheduling Dial-a-Ride Vehicles. *ORSA/TIMS National Meeting*, San Diego, California.

J. JAW, A. ODONI, H. PSARAFTIS and N. WILSON (1982) A Heuristic Algorithm for the Multi-Vehicle Many-to-Many Advance-Request Dial-A-Ride Problem. Working Paper MIT-UMTA-82-3, M.I.T., Cambridge, Massachusetts 02139.

J. JAW, A. ODONI, H. PSARAFTIS and N. WILSON (1986) A Heuristic Algorithm for the Multi-Vehicle Advance-Request Dial-A-Ride Problem with Time Windows. *Transportation Research B* 20, 243–257.

K. JORNSTEN, O. MADSEN and B. SORENSEN (1986) Exact Solution of the Vehicle Routing and Scheduling Problem with Time Windows by Variable Splitting. Working Paper, IMSOR, Denmark.

K. KNIGHT and J. HOFER (1968) Vehicle Scheduling with Timed and Connected Calls: A Case Study. *Operational Research Quarterly* 19, 299–310.

A. KOLEN, A. RINNOOY KAN and H. TRIENEKENS (1987) Vehicle Routing Time Windows. *Operations Research* 35, 266–273.

G. LAPORTE and Y. NOBERT (1987) Exact Algorithms for the Vehicle Routing Problem. *Annals of Discrete Mathematics* 31, 147–184.

J. K. LENSTRA, A. H. G. RINNOOY KAN (1981) Complexity of Vehicle Routing and Scheduling Problems. *Networks* 11, 221–227.

A. LEVIN (1971) Scheduling and Fleet Routing Models for Transportation Systems. *Transportation Science* 5, 232–255.

S. LIN (1965) Computer Solutions of the Traveling Salesman Problem. *Bell System Technical Journal* 44, 2245–2269.

O. MADSEN (1976) Optimal Scheduling of Trucks - A Routing Problem with Tight Due Times for Delivery. In *Optimization Applied to Transportation*

Systems, H. Strobel, R. Genser et M. Etchmaier (eds.), IIASA, Laxenburgh, Austria. 126–136.

T. MAGNANTI (1981) Combinatorial Optimization and Vehicle Fleet Planning: Perspectives and Prospects. *Networks* 11, 179–214.

G. MARTIN (1981) Aircraft Scheduling Considered as an N-Task, M-Parallel Machine Problem with Start-Times and Deadlines. *INFOR* 19, 152–161.

C. MILLER, A. TUCKER and R. ZEMLIN (1960) Integer Programming Formulations and Traveling Salesman Problems. *J.A.C.M.* 7, 326–329.

R. MOLE and S. JAMESON (1976) A Sequential Route Building Algorithm Employing a Generalized Savings Criterion. *Operational Research Quarterly* 27, 503–511.

I. OR (1976) Traveling Salesman-Type Combinatorial Problems and Their Relation to the Logistics of Blood Banking. *Ph. D. thesis, Department of Industrial Engineering and Management Sciences, Northwestern University.*

C. ORLOFF(1976) Route Constrained Fleet Scheduling. *Transportation Science* 10, 149–168.

H. PSARAFTIS (1980) A Dynamic Programming Solution to the Single-Vehicle, Many-to-Many, Immediate Request Dial-A-Ride Problem. *Transportation Science* 14, 130–154.

H. PSARAFTIS (1983a) An Exact Algorithm for the Single-Vehicle, Many-to-Many Dial-A-Ride Problem with Time Windows. *Transportation Science* 17, 351–357.

H. PSARAFTIS (1983b) K-Interchange Procedures for Local Search in a Precedence Constrained Routing Problem. *European Journal of Operational Research* 13, 517–524.

H. PSARAFTIS (1983c) Analysis of an $O(N^2)$ Heuristic for the Single Vehicle

Many-to-Many Euclidean Dial-A-Ride Problem. *Transportation Research* 17B, 133–145.

H. PULLEN and M. WEBB (1967) A Computer Application to a Transport Scheduling Problem. *Computer Journal* 10, 10–13.

S. ROY, J.-M. ROUSSEAU, G. LAPALME and J.A. FERLAND (1984a) Routing and Scheduling for the Transportation of Disabled Persons - The Algorithm. Working Paper TP 5596E, Transport Canada, Transport Development Center, Montreal, Canada.

S. ROY, J.-M. ROUSSEAU, G. LAPALME and J.A. FERLAND (1984b) Routing and Scheduling for the Transportation of Disabled Persons - The Tests. Working Paper TP 5598E, Transport Canada, Transport Development Center, Montreal, Canada.

M. SAVELSBERGH (1986) Local Search in Routing Problems with Time Windows. *Annals of Operation Research* 4, 285–305.

T. SEXTON and L. BODIN (1985a) Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: I. Scheduling. *Transportation Science* 19, 378–410.

T. SEXTON and L. BODIN (1985b) Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: II. Routing. *Transportation Science* 19, 411–435.

T. SEXTON and Y. CHOI (1986) Pick-up and Delivery of Partial Loads with Time Windows. *American Journal of Mathematical and Management Sciences* 6, 369–398.

M. SOLOMON (1983) The Vehicle Routing and Scheduling Problem with Time Window Constraints: Models and Algorithms. *Ph. D. thesis, Department of Decision Sciences, University of Pennsylvania.*

M. SOLOMON (1987) Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraints. *Operations Research* 35, 254–265.

M. SOLOMON (1986) On the Worst-Case Performance of Some Heuristics for the Vehicle Routing and Scheduling Problem with Time Window Constraints. *Networks* 16, 161–174.

M. SOLOMON and J. DESROSIERS (1988) Time Window Constrained Routing and Scheduling Problems. *Transportation Sciences* 22, 1–13.

F. SOUMIS, J. DESROSIERS and M. DESROCHERS (1985) Optimal Urban-Bus Routing with Scheduling Flexibilities. In *Lecture Notes in Control and Information Sciences* 59, Springer-Verlag 155–165.

A. SWERSEY and W. BALLARD (1983) Scheduling School Buses. *Management Science* 30, 844–853.

N. WILSON, J. SUSSMAN, H. WANG and B. HIGONNET (1971) Scheduling Algorithms for Dial-a-Ride Systems. Urban Systems Laboratory Report USL TR-70-13, MIT.

H. WILSON and H. WEISSBERG (1976) Advanced Dial-a-Ride Algorithms Research Project: Final Report. Report R76-20, Dept. of Civil Engineering, MIT.

H. WILSON and N. COLVIN (1977) Computer Control of the Rochester Dial-a-Ride System. Report R77-31, Dept. of Civil Engineering, MIT.

2

Répartition des requêtes entre les véhicules

2.1 Sommaire

Dans ce chapitre, nous proposons une approche pour la phase de répartition des requêtes entre les véhicules. Les contributions majeures de ce chapitre sont: (1) le concept innovateur de tâche ("mini-cluster") et (2) la généralisation de l'algorithme de génération de colonnes pour le problème de voyageur de commerce multiple avec fenêtre horaire pour résoudre les problèmes avec plusieurs dépots et des véhicules de caractéristiques différentes.

L'approche proposée se divise en deux étapes. Dans la première étape intervient l'important concept de tâche. Une procédure prend des décisions locales sur le regroupement et l'ordonnancement des requêtes de façon à regrouper dans des portions d'itinéraires faisables des requêtes voisines (de 3 à 4 requêtes en moyenne). Chacune des portions d'itinéraire forme ainsi une tâche. Les propriétés recherchées de ces tâches sont: (1) la faisabilité: la portion d'itinéraire de chaque tâche doit satisfaire l'ensemble des requêtes de la tâche en respectant les contraintes de fenêtres de temps, de couplage, de priorité et de capacités du véhicule, (2) l'efficacité: la somme des temps de déplacement des véhicules à l'intérieur des tâches est assez proche du minimum. Par exemple, si $+i$ repésente l'embarquement du client de la requête i et $-i$ son débarquement alors les portions d'itinéraires suivantes sont des tâches: $+1 \rightarrow -1$, $+1 \rightarrow +2 \rightarrow -2 \rightarrow -1$ et $+1 \rightarrow +2 \rightarrow -2 \rightarrow +3 \rightarrow -1 \rightarrow -3$ mais pas ceux-ci: $+1 \rightarrow +3 \rightarrow -1$ et $-1 \rightarrow +1$ puisque les contraintes de priorité et de couplage ne sont pas respectées. Plusieurs procédures heuristiques peuvent être utilisées pour créer un ensemble de tâches satisfaisant les propriétés ci-haut mentionnées. Par exemple, les requêtes pourraient être tout simplement regroupées selon des critères de proximité et de faisabilité. Nous avons plutôt utilisé une procédure qui effectue le morcellement des itinéraires d'une solution initiale aux endroits où le véhicule se déplace sans passagers. Il est facile de ce convaincre que les tâches ainsi créées ont la propriété de faisabilité. Cette solution initiale est une solution générée par l'algorithmes de Roy et al. (1984a).

Dans la deuxième étape, on répartit et ordonne les tâches entre les véhicules en solutionnant un problème de voyageur de commerce multiple avec fenêtres

de temps (m-TSPTW) et contraintes de compatibilité entre les véhicules et les tâches. A ce niveau les décisions globales sur la répartition sont effectuées en prenant en considération les coûts de répartition, les coûts d'ordonnancement ainsi que la faisabilité des phases suivantes. Desrosiers et al. (1984) ont proposé une approche optimale pour le m-TSPTW avec un seul type de véhicules; cette approche peut traiter rapidement des problèmes d'une centaine de tâches. L'approche est généralisée dans ce chapitre pour considérer une flotte mixte de véhicules avec plusieurs dépots ainsi que des contraintes de compatibilité entre les véhicules et les tâches. La généralisation théorique est simple par contre la géneralisation pratique ne l'est pas. Traiter 40 sous-problèmes efficacement n'est pas évident. Cette approche génère des itinéraires sur les tâches et assurent l'existence d'itinéraires et d'horaires faisables (sur les requêtes) pour chaque véhicule.

Nous avons publié cette approche dans "*Lecture Notes in Economics and Mathematical System 308: Computer-Aided Transit Scheduling*, J.R. Daduna, A. Wren (Editor), Springer-Verlag" en 1988, sous le titre "The Multiple Vehicle Dial-a-Ride Problem". Nous présentons cet article sous sa version TEXpuisque la typographie de la version publié est un peu déficiente.

La première section décrit le problème qui est celui du m-TSPTW avec contraintes de compatibilité entre les véhicules et les tâches.

La deuxième section présente la revue de la littérature. Cette revue est un condensé d'une partie de celle présentée au chapitre 1.

La troisième section expose la méthodologie proposée. Cette méthodologie diffère de celle que nous présentons au chapitre 5. Elle ne comprend pas la décomposition géographique ayant été développée pour résoudre des problèmes de très grande taille.

La quatrième section propose une façon d'obtenir un ensemble de tâches. Après la réoptimisation des itinéraires initiaux à l'aide de l'algorithme de 1-PDPTW présenté au chapitre 3 de cette thèse, chaque itinéraire est morcelé de la façon présentée ci-haut.

La cinquième section présente l'approche de génération de colonnes pour résoudre le m-TSPTW. Sa formulation s'étend à plusieurs types de véhicules et utilise une notation un peu différente de celle exposée au chapitre 1. La deuxième sous-section donne une formulation, en termes de tournées de véhicules, et décrit l'approche de génération de colonnes basée sur cette seconde formulation. Elle correspond à un problème de partitionnement avec contraintes de disponibilité des véhicules. La troisième sous-section montre comment obtenir une solution entière: une coupe sur le nombre de véhicules, un branchement sur les types de véhicules et un branchement sur les variables de flot binaires sont utilisés. La dernière sous-section présente un découpage temporel du problème. Ce découpage est efficace pour des problèmes de taille réduite. Pour des plus gros problèmes, un découpage temporel et géographique est présenté au chapitre 5.

Finalement, la sixième section de cet article présente les tests numériques. L'approche est testée sur 6 problèmes variant de 86 demandes à 880 demandes (36 tâches à 282 tâches). Sur le plus gros problème, des gains de 9 % sur le nombre de véhicules utilisés et de 18 % sur le déplacement ont été enregistrés par rapport à la solution initiale. Les résultats obtenus sont très encourageants. Sur les trois problèmes où le découpage temporel a été comparé à la résolution sans découpage, le nombre de véhicules utilisés est le même et la différence en temps de déplacement est inférieure à 2 %.

THE MULTIPLE VEHICLE DIAL-A-RIDE PROBLEM

by

JACQUES DESROSIERS
École des Hautes Études Commerciales
Montréal, Québec Canada H3T 1V6

YVAN DUMAS
Centre de recherche sur les transports
Montréal, Québec Canada H3J 3J7

FRANCOIS SOUMIS
École Polytechnique
Montréal, Québec Canada H3C 3A7

ABSTRACT

The multiple vehicle many-to-many routing problem is presented in the context of a dial-a-ride system. It is solved by mini-clustering first and optimal routing second. A heuristic mini-cluster algorithm groups together nearby customers who can be efficiently served by the same vehicle route segment. An optimal column generation algorithm then constructs routes corresponding to drivers' workdays by stringing together these vehicle route segments. Problems with up to 200 customers and 85 mini-clusters are easily solved. Larger problems are solved by dividing the day into time slices and applying the algorithm several times.

Key words: dial-a-ride, routing, scheduling, column generation, clustering.

1. PROBLEM DESCRIPTION

This article examines the use of mathematical optimization algorithms to construct vehicle routes in a large scale multi-vehicle many-to-many system in the context of the transportation of the handicapped. The main contributions of this paper are the innovative concept of the mini-clusters, the generalisation of the column generation algorithm for the m-TSP with time windows (Desrosiers, Soumis and Desrochers [1984]) to solve problems with several depots and availability constraints, and in addition, a method of decomposition into time slices to handle very large problems.

For a given day, we have a set of transportation requests, a homogeneous vehicle fleet and a set of pieces of work for drivers. Each transportation request involves: a pick-up stop and a delivery stop, a given number of clients to be transported, a desired pick-up or delivery time, a departure time interval and an arrival time interval, loading and unloading durations.

Vehicles have capacity limits on the number of wheelchairs, and an independent capacity for ambulatory passengers. A piece of work is part or all of a driver workday during which a driver and a vehicle are available to carry out circuits beginning and ending at the same depot. The types of pieces of work available, the number of each type, and the assignment of drivers to vehicles is determined in advance. We consider the case with several depots.

Distances and times between the various stops are given. The problem consists of minimizing the number of pieces of work used and then for this number of pieces of work, minimizing travel time (a function of the distance traveled) while satisfying the following constraints: **visiting constraints** (each pick-up and delivery stop is visited only once); **time window constraints** (the time constraints must be satisfied at each stop); **capacity constraints** (vehicle capacity cannot be exceeded); **depot constraints** (after a working period, each vehicle returns to the depot from which it departed); **pairing constraints** (for a given request, the same vehicle must visit the pick-up and delivery stops); **precedence constraints** (each client must be picked up before being delivered); **availability constraints** (the availability of driver pieces of work must be respected).

2. LITERATURE

The only known exact algorithm for the multiple vehicle routing problem with both pick-up and delivery in the presence of time window constraints has been developed by Dumas [1985]. This column generation algorithm has been applied to small problems (up to 50 requests) in the context of goods transportation with tight vehicle capacity constraints.

Bodin et al. [1983] in their state of the art article on vehicle routing and scheduling problems mention three algorithms for the multiple vehicle dial-a-ride problem. The NEIGUT/NBS procedure schedules one vehicle at a time and determines the best customer to be inserted into the route. The Jaw et al. [1986] procedure is a concurrent procedure. The day is split a priori into time slices of 20-30 minutes in length. Assuming that partial vehicle schedules exist over all preceding periods, this procedure forms clusters of customers for each of the vehicles in the given period. The Bodin and Sexton [1986] procedure is a traditional "cluster first, route second" approach. For a fixed fleet size, it partitions the set of requests into vehicle clusters and solves the resulting single vehicle dial-a-ride problems using a heuristic based on Benders' decomposition (Sexton and Bodin [1985a, 1985b]). Another concurrent insertion procedure using proximity measures has been developed by Roy et al. [1984a, 1984b].

The single vehicle dial-a-ride problem with time windows has also been studied by Psaraftis [1982] and Desrosiers, Dumas and Soumis [1986]. The interested reader should also refer to the work of Stein [1978], Armstrong and Garfinkel [1982], and Psaraftis [1980, 1983] on single and multiple vehicle dial-a-ride problems, to Desrosiers et al. [1985], Kolen et al. [1985], Savelsbergh [1984], Sorensen [1986], Solomon [1986a, 1986b], Swersey and Ballard [1984], and finally to the recent surveys of Desrochers et al. [1987] and Solomon and Desrosiers [1987] for other variants of vehicle routing problem with time window constraints.

3. PROPOSED METHODOLOGY

In a methodology of the "cluster first, route second" type (see Bodin et al. [1983]), a cluster is formed from the set of customers assigned to a vehicle, and routing is carried out separately for each vehicle. The routing problems are usually fairly easy to solve. However, the most important decisions are taken at the clustering stage and it is very difficult to globally construct a good set of clusters. For the multiple vehicle dial-a-ride problem, we therefore propose a "cluster first, route second" approach while moving a part of the clustering problem into the routing problem.

At the first stage, rather than defining clusters corresponding to a full day's work for one vehicle, we define **mini-clusters** grouping together nearby customers who can be transported by the same vehicle over a route segment. This grouping into mini-clusters of similar requests deals with local temporal and spatial considerations only. At the second stage, routes for all the vehicles are constructed simultaneously. This step deals with global considerations in transportation planning by assigning mini-clusters to vehicles.

This "mini-cluster first, route second" methodology is adopted in this paper to reoptimize an initial feasible solution (given by the Roy et al. [1984a] algorithm). The methodology uses four steps: steps 1A and 1B carry out the mini-clustering, while the routing is carried out in steps 2A and 2B.

1A. A single vehicle algorithm reoptimize each of the routes in the initial solution.

1B. These routes are divided into mini-clusters of customers.

2A. An integer linear programming with column generation is used to simultaneously reallocate mini-clusters between vehicles so as to minimize the number of pieces of work and the travel time.

2B. The mini-clusters are opened up and each route is reoptimized.

4. THE MINI-CLUSTERING STAGE

4.1 The single vehicle dial-a-ride algorithm

Steps 1A and 2B of our procedure require the optimization of routes one by one. This is carried out using a single vehicle dial-a-ride algorithm (Desrosiers, Dumas and Soumis [1986]). A route (cluster) usually serves from 5 to 20 clients i.e. the vehicle visits from 10 to 40 stops. The optimal route which minimizes the travel cost by a single vehicle visiting all stops is obtained by dynamic programming. It should be stressed that in this problem with a single vehicle, the pairing constraints are automatically satisfied and a piece of work is available (supplied by the initial feasible solution). The efficiency of our dynamic programming algorithm is due to the use of criteria for the elimination of states which do not or will not satisfy the time window, capacity and precedence constraints. Experiences with real data have shown that the execution time of this dynamic programming algorithm is a linear function of the number of stops.

4.2 Construction of mini-clusters

After optimizing the total travel time on a route, the route is divided into segments each time the vehicle becomes empty. The vehicle is empty at the beginning and at the end of the segment only. The customers transported on each segment make up a mini-cluster. Each mini-cluster groups together on average less than 4 requests for clients traveling at the same time in the same vehicle. Within a mini-cluster, the time window, precedence, pairing and capacity constraints are all satisfied. A time window giving the earliest and latest starting times is then associated with each mini-cluster based on the route segment and the time windows of the pick-up and delivery stops within the mini-cluster. The problem to be solved at step 2A becomes a routing problem with time window constraints on the set M of mini-clusters and availability constraints on the pieces of work.

5. ROUTING AND SCHEDULING ON MINI-CLUSTERS

The problem in step 2A involves mini-clusters $i \in M$ with a duration (total time between the first and last stop in the mini-cluster) and a cost (travel time between the first and last stop in the mini-cluster) as well as a time window $[a_i, b_i]$ on the visit starting time. We also have for each type of piece of work $k \in K$, an associated vehicle of type k, a departure point s_k, an arrival point e_k, a work period $[a^k, b^k]$ and a number of drivers available Q_k.

5.1 Mathematical formulation

The problem consists of finding vehicle schedules and routes which minimize the number of pieces of work and the travel time. In the formulation, we use a network structure $(N; A)$:

the set of nodes is $N = \cup_{k \in K} N^k$ with $N^k = M \cup \{s_k, e_k\}$;
the set of arcs is $A = \cup_{k \in K} A^k$ with $A^k = I \cup \{s_k\} \times M \cup M \times \{e_k\}$.

An arc (i, j) in the set $I \subseteq M^2$ is an unproductive (i.e. empty) run carried out by a vehicle as it goes from the end of mini-cluster i to the beginning of mini-cluster j. The time value t_{ij} is composed of the travel time on arc (i, j) plus the total time of minicluster i (which includes travel, waiting, loading and unloading times). An arc $(i, j) \in A$ is defined if time windows permit it (i.e. if $a_i + t_{ij} \leq b_j$). The cost value c_{ij} is a function of the distance traveled: it is given by the travel time on arc (i, j) plus the transportation (travel) time on mini-cluster i. Finally, a large fixed cost Γ is added to the cost of the arcs of the subsets $\{s_k\} \times M$, for all $k \in K$, to ensure that the number of pieces of work is minimized.

Two types of variables are used in the formulation: **flow variables** X_{ij}^k which are binary (where $X_{ij}^k = 1$ indicates that a vehicle of type k which is associated with the piece of work k travels from node i to node j), and continuous **time variables** T_i, $i \in M$ associated with the visit starting times for the mini-

clusters. In the objective function, costs are defined for all $k \in K$ as

$$\begin{aligned} c^k_{ij} &= c_{ij} && (i,j) \in I \\ c^k_{s_k j} &= \Gamma + c_{s_k j} && j \in M \\ c^k_{ie_k} &= c_{ie_k} && i \in M. \end{aligned}$$

Optimal routes respecting the scheduling and availability constraints are given by the solution of the following integer non-linear programming problem:

$$\min \sum_{k \in K} \sum_{(i,j) \in A^k} c^k_{ij} X^k_{ij} \tag{1}$$

subject to:

$$\sum_{k \in K} \sum_{j \mid (i,j) \in A^k} X^k_{ij} = 1 \qquad i \in M \tag{2}$$

$$\sum_{j \mid (\ell,j) \in A^k} X^k_{\ell j} = \sum_{i \mid (i,\ell) \in A^k} X^k_{i\ell} \qquad \ell \in M,\ k \in K \tag{3}$$

$$\sum_{j \in N^k} X^k_{s_k j} = \sum_{i \in N^k} X^k_{ie_k} \qquad k \in K \tag{4}$$

$$\sum_{j \in N^k} X^k_{s_k j} \leq Q_k \qquad k \in K \tag{5}$$

$$a_i \leq T_i \leq b_i \qquad i \in M \tag{6}$$

$$X^k_{ij} = 1 \Longrightarrow T_i + t_{ij} \leq T_j \qquad (i,j) \in I,\ k \in K \tag{7}$$

$$X^k_{s_k j} = 1 \Longrightarrow a^k + t_{s_k j} \leq T_j \qquad j \in M,\ k \in K \tag{8}$$

$$X^k_{ie_k} = 1 \Longrightarrow T_i + t_{ie_k} \leq b^k \qquad i \in M,\ k \in K \tag{9}$$

$$X^k_{ij} \quad \text{binary} \qquad (i,j) \in A^k,\ k \in K. \tag{10}$$

Constraints (2–3) ensure that one and the same type of vehicle enters and leaves each mini-cluster. Constraints (4) ensure that the same number of vehicles of type k begin and terminate each type of piece of work. Once variables X^k_{ij} for fixed k are given, it is easy to construct a schedule for each individual vehicle of type k. Inequalities (5) model the availability limitations on the number of pieces of work. Constraints (6) establish the time intervals on the visit starting time of mini-clusters while constraints (7–9) describe the compatibility requirements between the routes and schedules.

Rather than solving the problem in terms of flow variables in the network $(N;\ A)$, we generalized the solution method presented in Desrosiers, Soumis and Desrochers [1984]. The problem is formulated in terms of binary variables on the use of feasible routes in graphs $(N^k;\ A^k)$ and solve it by column generation. The new algorithm can solve problems with several depots and availability constraints.

5.2 Column generation scheme

The formulation in terms of feasible routes is a set partitioning problem with availability constraints on the pieces of work. We introduce the following notation:

Ω_k : the set of routes satisfying the scheduling constraints of piece of work k;

Y_r^k : binary variable, 1 if route r of piece of work k is used, 0 otherwise;

C_r^k : cost of route r in piece of work k (the sum of costs of arcs making up the route);

δ_{ri}^k : binary constant, 1 if route r of piece of work k goes through the mini-cluster i, 0 otherwise.

The master problem is formulated as follows:

$$\min \sum_{k\in K} \sum_{r\in\Omega_k} C_r^k Y_r^k \tag{11}$$

subject to:

$$\sum_{k\in K} \sum_{r\in\Omega_k} \delta_{ri}^k Y_r^k = 1 \qquad i \in M \tag{12}$$

$$\sum_{r\in\Omega_k} Y_r^k \leq Q_k \qquad k \in K \tag{13}$$

$$Y_r^k \quad \text{binary} \qquad r \in \Omega_k,\ k \in K. \tag{14}$$

The linear relaxation of the master problem is solved using the simplex algorithm. This method gives the dual variables σ_i, $i \in M$ and π_k, $k \in K$ necessary for column generation and lends itself well to reoptimization each time new columns are generated. The integer solution of the master is described in section 5.3.

A route of minimal marginal cost for piece of work k is generated by solving the shortest path subproblem with time window constraints on the nodes of the network $(N^k; A^k)$. Costs are given by

$$\bar{c}_{ij}^k = \begin{cases} c_{ij}^k - \pi_k & i = s_k \\ c_{ij}^k - \sigma_i & i \in M. \end{cases}$$

This constrained shortest path problem has been extensively studied and several algorithms have been designed: a Ford-Bellman type algorithm by Desrosiers, Pelletier and Soumis [1984], a Dijkstra type algorithm by Desrochers and Soumis [1985a], and a primal-dual reoptimization algorithm by Desrochers and Soumis [1985b]. The reader is referred to these publications for a full description of the dynamic programming algorithms.

5.3 Obtaining integer solutions

The sum of flows on the exit arcs from all s_k, $k \in K$ towards all mini-clusters $i \in M$ gives the number of pieces of work used. In our application, the fixed cost Γ is sufficiently large so that it is never worthwhile to increase the number of pieces of work in order to reduce travel time. When the total number of pieces of work used, given by $\sum_{k \in K} \sum_{r \in \Omega_k} Y_r^k = p$ is non-integer in the optimal linear solution of the master problem, a global cut is added to round this number up to at least the next integer:

$$\sum_{k \in K} \sum_{r \in \Omega_k} Y_r^k \geq \lceil p \rceil. \tag{15}$$

The master problem is then reoptimized by the dual simplex algorithm and new columns are generated as needed. For each subproblem $k \in K$, the

objective function is modified by adding the dual variable α associated with this new constraint (15):

$$\min \quad -\alpha + \sum_{(i,j)\in A^k} \bar{c}_{ij}^k X_{ij}^k. \tag{16}$$

If the solution is still non-integer, we use a branch-and-bound tree. A branching related to the use of pieces of work is the first priority. If $\sum_{r\in\Omega} Y_r^k = q_k$ is fractional, two branches are created in the tree. In each branch, a new constraint is added:

$$\sum_{r\in\Omega_k} Y_r^k \leq \lfloor q_k \rfloor \quad \text{on one branch and} \quad \sum_{r\in\Omega_k} \geq \lceil q_k \rceil \quad \text{on the other.}$$

New columns are then generated as required using all the subproblems, with the dual variable β_k associated with the new constraint transferred to the subproblem k. For each subproblem $k \in K$, the objective function is given by

$$\min -\alpha - \beta_k + \sum_{(i,j)\in A^k} \bar{c}_{ij}^k X_{ij}^k. \tag{17}$$

After branching on pieces of work, if the solution is still non-integer, branching is carried out on the binary variables Y_r^k. This is not done by fixing a Y_r^k variable at 0 or 1, but by setting at 0 or 1 the variables X_{ij}^k associated with the arcs which make up the route (see Desrosiers, Soumis and Desrochers [1984]). Exploration of the tree terminates when the number of pieces of work is optimal and when the gap with respect to the continuous simplex optimal travel time is less than 1%.

5.4 Decomposition on overlapping time slices for large problems

When the routing problem on mini-clusters becomes too large, the day is divided into successive overlapping time slices and a sequence of smaller problems is solved. The division of the day into time slices is carried out according to two criteria: an overlap time interval of 20 to 30 minutes between successive slices, and a number of mini-clusters varying between 30 and 40 per slice (including the mini-clusters in the overlap time). The reduced size of the problems solved for each time slice permits rapid handling by the column generation algorithm.

At the beginning of the algorithm, we have an initial feasible solution (steps 1A and 1B). For each route in this solution, we have pick-up and delivery stops of the network before, during and after the time slice being considered. For a given route, the associated piece of work is replaced by a piece of work of reduced time length. The starting place and time for the piece of work are defined as the last stop and time which the driver completed prior to the beginning of the time slice. Similarly, the finishing place and time are defined as the first stop and time to be visited after the end of the time slice. Pieces of work which were previously identical, are now different. In the column generation scheme, we have a subproblem for each different piece of work: the number of subproblems in a slice is equal to the number of routes of the feasible solution which covers the mini-clusters in this slice (each defining a new piece of work), plus the number of different unused pieces of work. This procedure retains after each time slice reoptimization a new feasible solution which is used to define the problem for the following time slice, and enables us to rapidly construct an initial basis.

For a given time slice and a given piece of work, if a vehicle comes from the depot, a fixed cost is assigned to the corresponding arc; if the vehicle is merely carrying out the continuation of piece of work, there is no fixed cost. To take account of this, the branching procedure on fractional variables was modified slightly to favour the use of columns with zero fixed cost.

6. TESTS

In the following sections, we present the numerical results for 6 problems

used to test the proposed methodology.

6.1 Test problems

The six test problems come from the Canadian cities of Sherbrooke (SHE), Toronto (TOR) and Montreal (MTL). The alphanumeric code identifying each problem indicates the city and the number of transportation requests. Table 1 describes the main characteristics of these six problems.

The number of requests varies from 86 to 190 for the four smallest problems and there are 410 and 880 requests for the two large Toronto problems. For each problem, the number of pick-up and delivery stops to be visited by all the vehicles is roughly twice the number of requests. This is because although there are a few common stops for the departure and arrival points of clients, additional stops for the depots associated with pieces of work have to be added. In the routing problem on mini-clusters, the number of nodes of the network $(N; A)$ to be visited is reduced by a factor of from 2.2 to 3.5.

Although the average number of requests making up a mini-cluster is less than 4, the distribution of the size of mini-clusters shows that there can be over 10 requests in some mini-clusters. The problem TOR185 handles morning requests only, while the other problems handle requests for a full day. Problem TOR410 includes regular requests in this city. Occasional requests are then added to form problem TOR880. These two problems amongst others were used for the evaluation of various transportation system scenarios for the handicapped by Bélisle et al. [1984].

Although the average duration of the arcs plus the duration of mini-clusters is from 6 to 8 times greater than the width of the time windows associated with the visit starting time of the mini-clusters, the network $(N; A)$ is not acyclical. For example, mini-clusters with a single client have time windows of from 30 to 45 minutes, but they can be of very short duration (5-10 minutes) and the network includes arcs of from 0 to 5 minutes.

Finally, the number of different pieces of work gives the number of subproblems generating feasible routes when minicluster routing is carried out with a single time slice problem (7 to 17 for the first 4 problems). When decomposition into several time slices is carried out, the number of subproblems per slice is roughly equal to the number of pieces of work used in the initial feasible solution (given by the Roy et al. [1984a] algorithm) plus the number of different unused pieces of work: this number is greater than 40 for problems TOR410 and TOR880.

6.2 Test results

The first four problems were solved without decomposition into time slices at the mini-cluster routing stage. Table 2 summarizes the results of the various steps in the procedure. The value of the objective function is given in two parts: the number of pieces of work, and the total travel time in minutes (a function of the distance traveled). In relation to the initial solution, the average reduction in travel time is given as follows: 3% after individual route optimization (step 1A), 7% after mini-cluster routing (step 2A), and 1% after the final optimization of each route (step 2B). For problems MTL158 and MTL190, the column generation algorithm also produced reductions of 4% and 9% respectively in the number of pieces of work used.

Test problems were solved on the University of Montreal Cyber 173 computer using the FORTRAN 5 compiler, and the simplex code comes from Land and Powell [1973]. In the computation time section of the table, note the superior performance of the optimization algorithms. A fairly large proportion of the total execution time is due to file management; further work is necessary to improve the efficiency of this component. Concerning the integer solution of the master problems at the column generation step, the integer optimal solution was obtained for problems SHE86 and MTL158 respectively at the first and second nodes of the enumeration tree. For problems TOR185 and MTL190, the integer solutions presented are optimal for the number of pieces of work and at 0.19% (6^{th} node) and 0.10% (3^{rd} node) from the continuous simplex solution for travel time.

Table 3 shows the results of problems solved by decomposition into over-

lapping time slices at the mini-cluster routing stage. For the first 3 problems, i.e. TOR185, MTL158 and MTL190, comparisons can be made with solutions obtained without decomposition. For these three problems, the same number of pieces of work is obtained. Travel times are identical for problem MTL158, and are slightly higher (by 1% and 2% respectively) for problems TOR185 and MTL190. Computation times for the column generation algorithm are lower for two out of three problems; however there is a slight increase in the time required for file management. Overall, solutions remain of about the same quality and the decomposition into time slices is an efficient alternative for the solution of large problems.

The two large problems TOR410 and TOR880 were both solved using this decomposition approach. Comparison with the initial solutions given by Roy et al. [1984a] algorithm shows that the number of pieces of work can be significantly reduced (by 19% and 9%), as can the total travel time (by 38% and 18%). Computation times for the optimization of routes one at a time by dynamic programming as described in Desrosiers, Dumas and Soumis [1986] are good. The 30 routes of problem TOR410 contain on average 13.5 requests while the 48 routes of problem TOR880 contain 18.3 requests on average.

The column generation algorithm (step 2A) still performs well. The average solution time for the 13 decomposed problems for problem TOR880 is 60 seconds. Each problem in the decomposition involves around 35 mini-clusters to be covered by over 40 different pieces of work i.e. over 40 different subproblems generating feasible routes. For the 5 problems solved by decomposition, the optimal integer solution was obtained for the smaller routing problems on mini-clusters 18 times out of 24. For the other 6 problems the gap with respect to the continuous simplex solution was 0.2% on average.

Finally, the total computation time of over 22 minutes for problem TOR880 may seem prohibitive. Recall that this problem is essentially a multiple traveling salesman problem with two depots and 1760 nodes to be visited subject to time window, capacity, precedence, pairing and availability constraints. We believe that the quality of the solution obtained is encouraging for the use of mathematical optimization techniques like those described in this article.

7. CONCLUSION

The problem of vehicle route construction for the transportation of the handicapped is difficult to solve, both because of the size of the problem and because of the multiplicity and complexity of the constraints. The construction of mini-clusters of clients traveling at the same time in the same vehicle lessens the difficulties by reducing the size of the new routing problem while satisfying in total or in part certain constraints (precedence, pairing, time windows). The new routing problem on the mini-clusters is however also difficult to solve and is subject to time window, availability and multiple depot constraints. But through a mathematical programming methodology using column generation, these constraints can be handled efficiently in subproblems generating feasible routes.

ACKNOWLEDGMENTS

The authors wish to thank the Federal Government of Canada (NSERC and PUT programs), the Quebec Government (FCAR program), and l'École des Hautes Études Commerciales de Montréal, for their financial support.

TABLE 1: TEST PROBLEM DESCRIPTION

IDENTIFICATION[a]	SHE86	TOR185	MTL158	MTL190	TOR410	TOR880
NUMBER OF MINI-CLUSTERS	36	56	66	85	115	282
NETWORK SIZE $(N;A)$[b]						
NODES	56	70	90	119	153	310
ARCS	672	906	1835	3145	4979	(c)
SIZE DISTRIBUTION OF MINI-CLUSTERS						
(1–4)	30	40	59	77	81	212
(5–10)	6	14	7	8	31	66
(11–14)	0	2	0	0	3	4
AVERAGE	2.4	3.3	2.4	2.2	3.5	3.1
VISIT STARTING TIME DISTRIBUTION OF MINI-CLUSTERS						
(5–10H)	11	56	27	29	54	105
(10–14H)	9	0	1	1	0	54
(14–18H)	13	0	37	43	61	93
(18–24H)	4	0	1	12	0	30
TIME WINDOWS ON MINI-CLUSTERS						
AVERAGE[d]	19	22	15	16	21	21
MAXIMUM	40	40	30	30	45	45
DURATION OF MINI-CLUSTERS						
AVERAGE[e]	33	60	41	40	65	60
MAXIMUM	113	192	128	128	195	235
DURATION OF ARCS						
AVERAGE[d]	39	66	57	58	80	83
MAXIMUM	126	226	182	182	243	291
DIFFERENT PIECES OF WORK	10	7	12	17	19	19
TOTAL AVAILABILITY	12	28	24	34	69	60

(a) code identifying the city and the number of requests
(b) the Toronto problems contain two different depots
(c) for this very large problem, solved by decomposition, the number of arcs is estimated to be over 30,000
(d) with a minimum of 0 minutes
(e) with a minimum of 5 minutes

TABLE 2: RESULTS OF PROBLEMS SOLVED WITHOUT TIME SLICE DECOMPOSITION AT THE MINI-CLUSTER ROUTING STAGE

IDENTIFICATION[a]	SHE86		TOR185		MTL158		MTL190	
INITIAL SOLUTION[b]	10	1158	26	4312	24	3359	34	4323
STEP 1A	10	1125	26	4101	24	3242	34	4202
STEP 2A	10	1084	26	3686	23	3094	31	3880
STEP 2B	10	1081	26	3614	23	3043	31	3801
GAIN	0%	7%	0%	16%	4%	9%	9%	12%
COMPUTATION TIME IN SECONDS ON A CDC 173 COMPUTER								
STEPS 1A AND 2B[c]	19	(20)	80	(52)	23	(47)	31	(65)
STEP 2A								
- SIMPLEX[d]	8	(2)*	41	(6)	36	(1)*	104	(3)
- SUBPROBLEMS[e]	3	(131)	31	(281)	33	(414)	54	(536)
FILE MANAGEMENT[f]	33		44		79		78	
TOTAL	63		196		171		267	

(a) code identifying the city and the number of requests
(b) number of pieces of work and total travel time in minutes
(c) computation time for the single vehicle dial-a-ride algorithm and, in brackets, the number of routes reoptimized
(d) computation time for the simplex algorithm and, in brackets, the number of nodes explored in the branch-and-bound; * for optimality
(e) computation time for the shortest path with time windows algorithm and, in brackets, the number of columns generated
(f) creation and update of files of routes, mini-clusters, pieces of work and time subtables

TABLE 3: RESULTS OF PROBLEMS SOLVED BY TIME SLICE DECOMPOSITION AT THE MINI-CLUSTER ROUTING STAGE

IDENTIFICATION[a]	TOR185		MTL158		MTL190		TOR410		TOR880	
INITIAL SOLUTION[b]	26	4312	24	3359	34	4323	37	8791	53	19273
SOLUTION WITHOUT DECOMPOSITION	26	3614	23	3043	31	3801		—		—
TIME SLICES		2		2		3		4		13
SOLUTION WITH DECOMPOSITION	26	3659	23	3043	31	3875	30	5455	48	15765
GAIN VS INITIAL SOLUTION	0%	15%	4%	9%	9%	10%	19%	38%	9%	18%
LOSS VS SOLUTION WITHOUT DECOMPOSITION	0%	1%	0%	0%	0%	2%		—		—

COMPUTATION TIMES IN SECONDS ON A CDC 173 COMPUTER					
STEPS 1A AND 2B[c]	83(52)	23(47)	30(65)	128(68)	256(101)
STEP 2A[d]	77	46	49	150	770
FILE MANAGEMENT[e]	55	90	102	123	279
TOTAL	215	159	181	401	1305

(a) code identifying the city and the number of requests

(b) number of pieces of work and total travel time in minutes

(c) computation time for the single vehicle dial-a-ride algorithm and, in brackets, the number of routes reoptimized

(d) total computation time for the simplex algorithm and for the shortest path with time window algorithm for all time slices

(e) creation and update of files of routes, mini-clusters, pieces of work and time subtables, for all time slices

REFERENCES

ARMSTRONG, G.R. and R.S. GARFINKEL. 1982. Dynamic Programming Solution of the Single and Multiple Pick-up and Delivery Problem with Application to Dial-a-Ride. Working Paper #162, College of Business Administration, University of Tennessee, Knoxville, Tennessee.

BELISLE, J.P., SOUMIS, F. ROY, S., DESROSIERS, J., DUMAS, Y. and J.M. ROUSSEAU. 1986. The Impact on Vehicle Routing of Various Operational Rules of a Transportation System for Handicapped Persons. Third International Conference on Mobility and Transport of Elderly and Handicapped Persons, Office of the Secretary of Transportation. Washington D. C. 20590, 6.47–6.50.

BODIN, L., B. GOLDEN, A. ASSAD and M. BALL. 1983. Routing and Scheduling of Vehicles and Crews – The State of the Art, *Computers & Operation Research* 10, 63–211.

BODIN, L. and T. SEXTON. 1986. The Multi-Vehicle Subscriber Dial-a-Ride Problem. The Delivery of Urban Services, *TIMS Studies in the Management Sciences* 22, 73–86.

DESROCHERS, M. and F. SOUMIS. 1985a. A Generalized Permanent Labelling Algorithm for the Shortest Path Problem with Time Windows. Publication #394A, Centre de recherche sur les transports, Université de Montréal, Montréal Canada H3J 3J7.

DESROCHERS, M. and F. SOUMIS. 1985b. A Reoptimization Algorithm for the Shortest Path Problem with Time Windows. Publication #397A, Centre de recherche sur les transports, Université de Montréal, Montréal Canada H3J 3J7.

DESROCHERS, M., J.K. LENSTRA, M.W.P. SAVELSBERGH and F. SOUMIS. 1987. Vehicle Routing with Time Windows: Optimization and Approximation. Forthcoming in *TIMS STUDIES in the Management Science.*

DESROSIERS, J., Y. DUMAS and F. SOUMIS. 1986. A Dynamic Programming Solution of the Large-Scale Single-Vehicle Dial-a-Ride Problem with Time Windows. *American Journal of Mathematical and Management Sciences*, 6, 301–325.

DESROSIERS, J., P. PELLETIER and F. SOUMIS. 1984. Plus court chemin avec contraintes d'horaires. *R.A.I.R.O. Recherche opérationnelle* 17(4), 357–377.

DESROSIERS, J., M. SAUVE and F. SOUMIS. 1985. Lagrangian Relaxation Methods for Solving the Minimum Fleet Size Multiple Traveling Salesman Problem with Time Windows. Publication #396, Centre de recherche sur les transports, Université de Montréal, Montréal Canada H3J 3J7.Forthcoming in *Management Science.*

DESROSIERS, J., F. SOUMIS and M. DESROCHERS. 1984. Routing with Time Windows by Column Generation. *Networks* 14, 545-565.

DUMAS, Y. 1985. Confection d'itinéraires de véhicules en vue du transport de plusieurs origines à plusieurs destinations. Publication #434, Centre de recherche sur les transports, Université de Montréal, Montréal Canada H3J 3J7.

JAW, J., A. ODONI, H. PSARAFTIS and N. WILSON. 1986. A Heuristic Algorithm for the Multi-Vehicle Advanced Request Dial-a-Ride Problem with Time Windows. *Transportation Research* 20B, 243–257.

KOLEN, A., A. RINNOOY KAN and H. TRIENEKENS. 1985. Vehicle Routing with Time Windows. Working Paper, Erasmus University, Rotterdam.

LAND, A. H. and S. POWELL. 1973. *Fortran Codes for Mathematical Programming: Linear, Quadratic and Discrete.* John Wiley & Sons, London.

PSARAFTIS, H. 1980. A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science* 14, 130–154.

PSARAFTIS, H. 1983. An Exact Algorithm for the Single Vehicle Many-to-Many Dial-a-Ride Problem with Time Windows. *Transportation Science* 17, 351–357.

PSARAFTIS, H. 1983. Analysis of an $O(N^2)$ Heuristic for the Single Vehicle Many-to-Many Euclidian Dial-a-Ride Problem. *Transportation Research*, 17B(2), 133–145.

ROY, S., J.M. ROUSSEAU, G. LAPALME and J.A. FERLAND. 1984a. Routing and Scheduling for the Transportation of Disabled Person – The Algorithm. TP 5596E, Transport Canada, Transport Development Centre, Montreal Canada H2Z 1X4.

ROY, S., J.M. ROUSSEAU, G. LAPALME and J.A. FERLAND. 1984b. Routing and Scheduling for the Transportation of Disabled Persons – The Tests. TP 5598E, Transport Canada, Transport Development Centre, Montreal Canada H2Z 1X4.

SAVELSBERGH, M. 1984. Local Search in Routing Problem with Time Windows. Report OS-R8409, Center for Mathematics and Computer Science, Amsterdam.

SEXTON, T. and L. BODIN. 1985a. Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: I. Scheduling. *Transportation Science* 19(4), 378–410.

SEXTON, T. and L. BODIN. 1985b. Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: II. Routing. *Transportation Science* 19(4), 411–435.

SOLOMON, M. 1986a. On the Worst-Case Performance of Some Heuristics for the Vehicle Routing and Scheduling Problem with Time Window Constraints.*Networks* 16, 161-174.

SOLOMON, M. 1986b. Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraints. Forthcoming in *Operations Research.*

SOLOMON, M. and J. DESROSIERS. 1987. Time Window Constrained Routing and Scheduling Problems: A Survey. Forthcoming in *Transportation Science.*

SORENSEN, B. 1986. Interactive Distribution Planning. Ph.D. Thesis, The Institute of Mathematical Statistics and Operations Research, DK-2800 Lyngby, Denmark.

STEIN, D. 1978. Scheduling Dial-a-Ride Transportation Systems. *Transportation Science* 12, 232–249.

SWERSEY,A.J. and W. BALLARD. 1984. Scheduling School Buses. *Management Science* 30, 844-853.

3

Optimisation de l'itinéraire d'un véhicule

3.1 Sommaire

Nous proposons, dans ce chapitre, un algorithme pour l'optimisation de l'itinéraire d'un véhicule. Cet algorithme est conçu pour résoudre le problème de cueillettes et livraisons par un seul véhicule, soit le 1-PDPTW où les contraintes de fenêtres horaires et/ou de capacité sont restrictives. La contribution majeure est le développement d'un algorithme exact de programmation dynamique vers l'avant permettant d'augmenter de 10 à 40 requêtes la taille des problèmes résolubles. Nos travaux ayant déjà fait l'objet d'une publication dans "*American Journal of Mathematical and Management Sciences* 6, pages 301–325" sous le titre "A Dynamic Programming Solution of the Large-Scale Single-Vehicle Dial-a-Ride Problem with Time Windows", nous vous en présentons ici sa version intégrale.

La section 1 de cet article introduit la revue de la littérature et la formulation mathématique du problème. La revue de la littérature est plus détaillée et plus spécifique que celle présentée à la section 1.4.5 de cette thèse et la formulation mathématique utilise une notation différente de celle déjà présentée. Ainsi, l'intervalle de temps $[a_i, b_i]$ est remplacé par $[A_i, B_i]$, la quantité à transporter d_i par C_i, le temps de service s_i par M_i, le coût de déplacement $c_{i,j}$ par $D_{i,j}$, le temps de déplacement $t_{i,j}$ par $T_{i,j}$ et finalement la variable de temps T_i est remplacée par t_i.

La section 2 de l'article propose la méthode de résolution. Il s'agit d'une méthode de programmation dynamique exécutée vers l'avant (forward). Un état est défini comme un couple constitué d'un ensemble de noeuds visités et du dernier de ces noeuds visité. Les étiquettes sont bidimentionnelles, chacune d'elle inclue le coût de déplacement et l'heure de service du dernier noeud visité. Puisqu'il y a plusieurs dimensions, l'équation de récurence prend une forme un peu complexe.

La section 3 présente différents critères d'élimination d'états. L'efficacité de notre algorithme de programmation dynamique est due, en grande partie, à l'utilisation de critères efficaces d'élimination des états. Ainsi, nous avons utilisé des critères basés sur la non post-faisabilité (i.e. basés sur l'élimination des états qui ne peuvent engendrer, dans les itérations ultérieures, une solution faisable)

ainsi que des critères d'élimination de la prolifération d'états dans le cas où plusieurs clients sont au même endroit. Les états infaisables, dominés ou ne pouvant pas faire partie d'une solution optimale sont éliminés. Les autres articles publiés n'ont pas exploité l'ensemble des contraintes restrictives, en particulier la contrainte de visiter tous les noeuds, qui permet l'utilisation de critères sur la non post-faisabilité.

La section 4 présente les résultats numériques obtenus. Elle inclue un exemple d'un problème avec 2 requêtes, quelques détails importants de l'implantation ainsi que les tests effectués sur trois villes différentes. L'analyse des résultats numériques démontre une grande efficacité de l'algorithme présenté. Des problèmes jusqu'à 40 requêtes (80 noeuds) ont été résolus en moins de 6 secondes sur un CYBER 173. En fait, les résultats montrent que le temps de résolution croit linéairement avec le nombre de requêtes. Notons finalement que l'algorithme est conçu pour être efficace sur des problèmes très constraint et qu'il utilise l'hypothèse que les temps de déplacement satisfont l'inégalité du triangle i.e. entre deux arrêts, le temps de déplacement du chemin direct est inférieur ou égal à tout autre chemin.

AMERICAN JOURNAL OF MATHEMATICAL AND MANAGEMENT SCIENCES
Copyright© 1986 by American Sciences Press, Inc.

A DYNAMIC PROGRAMMING SOLUTION OF THE LARGE-SCALE SINGLE-VEHICLE DIAL-A-RIDE PROBLEM WITH TIME WINDOWS

Jacques Desrosiers
École des Hautes Études Commerciales de Montréal
Montréal, Canada H3T 1V6

Yvan Dumas
Centre de recherche sur les transports, Université de Montréal
Montréal, Canada H3C 3J7

François Soumis
École Polytechnique de Montréal
Montréal, Canada H3C 3A7

SYNOPTIC ABSTRACT

The single-vehicle dial-a-ride problem with time window constraints for both pick-up and delivery locations, and precedence and capacity constraints, is solved using a forward dynamic programming algorithm. The total distance is minimized. The development of criteria for the elimination of infeasible states results in solution times which increase linearly with problem size.

Key Words and Phrases: dial-a-ride; dynamic programming; routing; scheduling

1986, VOL. 6, NOS. 3 & 4, 301-325
0196-6324/86/030301-25 $8.00

1. INTRODUCTION.

The problem examined in this article is the Single-Vehicle Dial-a-Ride Problem. In this problem, a vehicle picks up riders at their place of origin and takes them to their destination, without exceeding vehicle capacity, and respecting time window constraints at both the origins and the destinations. The objective is to find the itinerary which minimizes the total distance traveled.

1.1. Literature review. The single-vehicle problem has been studied by Sexton and Bodin (1985a, 1985b) who investigated the dial-a-ride problem in which each customer specifies a desired time for pick-up or delivery. The result is a heuristic routing and scheduling algorithm, based on Benders' decomposition, which minimizes total customer inconvenience and is known to produce high quality solutions. Psaraftis (1980) developed a backward dynamic programming method for the single-vehicle problem, but rather than imposing time window constraints at the origins and destinations, he imposed a maximum position shift with the respect to the ordering of pick-up and delivery times requested. Computation time for n customers is $O(n^2 3^n)$. Armstrong and Garfinkel (1982) also developed a backward recursion procedure for this problem using a maximum position shift formulation rather than time windows; their procedure is much more powerful for heavily constrained problems. It allows a great reduction in the number of states generated, but the algorithm cannot optimally solve the case where clock time is used to define window constraints. Psaraftis (1982) developed a forward dynamic programming method with true clock time windows at both pick-up and delivery locations. His objective function minimizes the total time needed to service all customers. Unfortunately, this algorithm is independent of the tightness of the time window constraints, and in practice it cannot solve problems with more than about eight to ten customers. Psaraftis (1983) also

analyzes an $O(n^2)$ heuristic for the single-vehicle problem.

Bodin, Golden, Assad and Ball (1983), in their state of the art article, describe three algorithms for the multi-vehicle problem : the NEIGUT/NBS algorithm (a sequential insertion procedure), the Jaw, Odoni, Psaraftis and Wilson (1981) algorithm (a concurrent procedure) and the Bodin and Sexton (1986) algorithm (a clustering/concurrent insertion procedure). Two other algorithms have recently been developped at the University of Montreal : the work of Roy, Rousseau, Lapalme and Ferland (1984a, 1984b), which is also based on a concurrent insertion procedure; and the algorithm of Desrosiers, Dumas and Soumis (1984), a "mini-cluster first, routing second" approach using a column generation scheme.

This paper presents an optimal solution to the single-vehicle problem with clock time windows using a forward dynamic programming approach that significantly reduces the number of states generated. We also use the two dimensional (time, cost) labeling introduced in Desrosiers, Pelletier and Soumis (1983) for solving the shortest path problem with time windows by dynamic programming. We minimize the total distance traveled and not the total time required to serve all customers. The algorithm proposed can efficiently solve problems of up to 40 service requests and is presently used to optimize the routes constructed in a large scale multi-vehicle problem : Bélisle, Soumis, Roy, Desrosiers, Dumas and Rousseau (1984) and Desrosiers, Dumas and Soumis (1984).

1.2. Formulation. Consider a set of n service requests. Associated with each request we have

1) an origin node i with a departure time interval $[A_i, B_i]$, an integer value $C_i > 1$ representing the number of people to be picked up and M_i, the time required to handle pick-up;

2) a destination node $n + i$ with an arrival time interval $[A_{n+i}, B_{n+i}]$, a value $C_{n+i} = -C_i$ representing the number of people to be dropped off and M_{n+i}, the drop-off time.

In addition to these 2n nodes, we have a node 0 to represent the vehicle departure point (with time interval $[A_0, B_0]$) and node $2n + 1$ for the vehicle arrival point (with time interval $[A_{2n+1}, B_{2n+1}]$). D_{ij} and T_{ij} respectively denote the distance traveled and the travel time between nodes i and j. The constant C represents the capacity limit of the vehicle.

The formulation includes <u>flow variables</u> x_{ij}, <u>time variables</u> t_i and <u>vehicle load variables</u> y_i, where

$$x_{ij} = \begin{cases} 1 & \text{if the route uses arc } (i,j) \\ 0 & \text{otherwise,} \end{cases}$$

$t_i = \{\text{arrival time at node i}\}$,

$y_i = \{\text{the number of passengers in the vehicle on its departure from node i}\}$.

The shortest route satisfying precedence constraints, time window constraints and vehicle capacity constraints is the solution of the problem :

$$\text{Min } Z = \sum_{i=0}^{2n} \sum_{j=1}^{2n+1} D_{ij} x_{ij} \qquad (1)$$

Subject to

$$\sum_{j=1}^{n} x_{0j} = 1 \qquad (2)$$

$$\sum_{i=n+1}^{2n} x_{i,2n+1} = 1 \qquad (3)$$

$$\sum_{i=0}^{2n} x_{ii} = \sum_{j=1}^{2n+1} x_{ij} = 1 \qquad i = 1,\ldots,2n \qquad (4)$$

$$x_{ij} \text{ binary} \qquad \begin{matrix} i = 0,1,\ldots,2n \\ j = 1,\ldots,2n+1 \end{matrix} \qquad (5)$$

$$A_i \leq t_i \leq B_i \qquad i = 0,1,\ldots,2n+1 \qquad (6)$$

$$t_i + M_i + T_{i,n+i} \leq t_{n+i} \qquad i = 1,\ldots,n \qquad (7)$$

$$x_{ij} = 1 \Rightarrow t_i + M_i + T_{ij} \leq t_j \qquad i = 0,1,\ldots,2n;\ j = 1,\ldots,2n+1 \qquad (8)$$

$$y_0 = 0 \ ; \ 0 \leq y_i \leq C \qquad i = 1,\ldots,2n+1 \qquad (9)$$

$$x_{ij} = 1 \Rightarrow y_i + C_j = y_j \qquad i = 0,1,\ldots,2;\ j = 1,\ldots,2n+1. \qquad (10)$$

This formulation includes a flow problem structure (1-5) where each node is visited only once (4) with a single vehicle departing from node 0 (2) and terminating at node 2n+1 (3). The problem also includes time window constraints (6), origin-destination precedence constraints (7), and vehicle capacity constraints (9). Constraints (8) describe the relation between the flow variables and the time variables, while constraints (10) describe the relation between the flow variables and vehicle load at each node. The problem is essentially a traveling salesman problem with additional constraints, with the non-linear constraints (8) ensuring sub-tour elimination. In fact, these constraints require increasing times at each of the nodes along the route, thus preventing sub-tour formation.

By minimizing the total distance traveled, riding time within the vehicle is not minimized in equation (8). This objective function is less general than others proposed for minimizing user inconvenience. Using the definition of time windows given in Roy, Rousseau, Lapalme and Ferland (1984), it is possible to take user inconvenience issues into account. For example, if a request i has a desired departure time A_i, the latest time the vehicle may arrive at the origin node is defined as $B_i = A_i + a$, with $a \leq 45$ minutes depending on the priority of the request. The lower and upper bounds of the time window at the destination node n + i are given by $A_{n+i} = A_i + M_i + T_{i,n+i}$ and $B_{n+i} = A_{n+i} + a + b$, with $b = 0.5\ T_{i,n+i}$ and where a+b is also

constrained to be at least 15 minutes but not more than 60 minutes.

By defining the time windows in this way, we respect the constraint on the maximum deviation from the desired service time (parameter a). Constraints on the maximum excess travel time and on maximum total travel time are relaxed, but are controlled by the sum of the parameters a+b. With this definition, the time windows are sufficiently large that the problem does not reduce to visiting a sequence of origin-destination pairs.

The solution method described in Section 2 is appropriate for any objective function which can be written as a linear function of the x_{ij} variables (distance and/or time).

2. SOLUTION METHOD.

Problem (1) subject to (2)-(10) is solved using a forward dynamic programming method. The vehicle is initially located at the departure node 0. At the first iteration, the states are made up of routes visiting a single node from among the origins. At each subsequent iteration k ($2 \leq k \leq 2n$), the states are constructed from the states of the previous iteration and are made up of routes visiting one additional node from among the origins and destinations. At the last iteration ($k=2n+1$), the vehicle must go to the arrival node $2n+1$. We have improved the efficiency of this method, even for large-scale problems, by eliminating states (Section 3) which are incompatible with vehicle capacity, precedence and time window constraints.

2.1 Definition of states. At iteration k ($1<k<2n$), we define a state (S,i) if, starting at node 0, there exists a feasible route which visits all the nodes in $S \subseteq \{1,\ldots,2n\}$ and terminates at node $i \in S$; S is a non-ordered set of cardinality k.

State (S,i) is ~~ente~~-feasible if there exists an order of visiting the nodes in S ending with node i which respects

vehicle capacity, precedence, and time window constraints. State (S,i) is post-feasible if, beginning at node i, there exists an order of visiting the nodes in $\bar{S}$ which respects the same constraints. In the proposed algorithm, we eliminate all states which are not ante-feasible. Only some of the non-post-feasible states are eliminated: we do not eliminate states which would be more costly to identify than the savings likely to be achieved by their elimination.

2.2. Definition of labels. For state (S,i), there are several routes from node 0 to node i. These routes have the same vehicle load at node i, regardless of the order in which the nodes of S are visited: this load is denoted by y(S). However, the routes differ in terms of the arrival time at node i and the distance traveled (cost) to reach this node, depending on the order in which the nodes are visited. A given route corresponding to state (S,i) is denoted by (S_α, i). For each route, we define a (time, cost) label as $(t(S_\alpha, i), z(S_\alpha, i))$, where $t(S_\alpha, i)$ and $z(S_\alpha, i)$ represent respectively the arrival time at node i, and the distance traveled to reach this node, upon visiting the nodes of S in the order described by S_α.

At state (S,i) only some of the labels (or routes) are stored. A label is eliminated if it cannot be part of the minimum cost route from 0 to 2n+1, using the following optimality principle. Let $R = S \cup \{j\}$ and let $(t(R_\alpha, j), z(R_\alpha, j))$ be the label associated with a route from 0 to j visiting the nodes of R. If this is a minimum cost route from among all routes arriving at node j at times $t_j \leq t(R_\alpha, j)$ and if the arc (i,j) is last in this route, then the sub-route from 0 to i is a minimum cost route from among all routes arriving at node i at times $t_i \leq t(R_\alpha, j) - (M_i + T_{ij})$.

It is therefore sufficient to store the labels associated with the minimum cost routes from 0 to i visiting the set S for each arrival time i. We may eliminate for state (S,i) the labels

dominated using the following partial ordering relation $\leq$ on the (time, cost) pairs :

$$(t,z) \leq (t',z') \Leftrightarrow t \leq t' \text{ and } z \leq z'. \qquad (11)$$

The label (t',z') is eliminated by (11) if there exists another label with both a lesser (or equal) time and a lesser (or equal) cost. This partial ordering relation reduces the size of the set of labels to be stored at each state (S,i). This set is given by:

$$H(S,i) = \{(t(S_\alpha,i),\ z(S_\alpha,i)),\ \alpha \in \Omega(S,i)\} \qquad (12)$$

where $\Omega(S,i)$ corresponds to the set of routes visiting the nodes in S and ending at node i, that are not eliminated.

The labels associated with a state are placed in a list with times in strictly increasing order. It can then be shown, based on the partial ordering relation, that the costs will be in strictly decreasing order. The efficient management of these lists is described in Desrosiers, Pelletier and Soumis (1983) for a dynamic programming algorithm applied to the shortest path problem with time window constraints on the nodes.

2.3. Recursion. The first iteration is carried out by visiting the origins from node 0 resulting in the set of states

$$\{(\{j\},j) \quad j \in \{1,\ldots,n\}\} \qquad (13)$$

For each state, information on the load, time and cost is easily obtained. The load is equal to the number of people to be picked up at the origin visited, and there is only one (time, cost) label associated with a state. Also

$$y(\{j\}) = C_j \qquad j \in \{1,\ldots,n\} \quad (14)$$

$$H(\{j\},j) = \{(\max [A_j,\ A_0 + T_{0j}],\ D_{0j}\} \qquad j \in \{1,\ldots,n\}. \quad (15)$$

At subsequent iterations, $2 \leq k \leq 2n$, new states are constructed by adding one node to the total visited at the preceding iteration. Let $(S \cup \{j\},j)$ be such a state with $j \notin S$.

The load at node j is given by

$$y(S \cup \{j\}) = y(S) + C_j. \tag{16}$$

To obtain the list of labels at state $(S \cup \{j\}, j)$, we must adjust the time and cost of each label in $H(S,i)$ for all states (S,i) which are used to construct state $(S \cup \{j\}, j)$ when node j is added to the state (S,i). The new set of labels is obtained from the previous one by the function

$$f_{ij}(\{(t_i, z_j)\}) = \begin{cases} \{(\max [A_j, t_i + M_i + T_{ij}], z_i + D_{ij})\} \\ \quad \text{if } t_i + M_i + T_{ij} < B_j \\ 0 \quad \text{otherwise} \end{cases} \tag{17}$$

A new (time, cost) label can be created if starting at node i at time t_i, it is possible to arrive at node j before time B_j (the upper bound of the time window at node j). From a given state (S,i), we obtain for state $(S \cup \{j\}, j)$ the set of labels

$$F_{ij}(H(S,i)) = \bigcup_{\alpha \in \Omega(S,i)} f_{ij}(\{(t(S_\alpha, i), z(S_\alpha, i))\}) \tag{18}$$

Finally, starting with all the states (S,i) which can be used to construct the new state $(S \cup \{j\}, j)$, we obtain a new list of labels:

$$H(S \cup \{j\}, j) = \cup F_{ij}(H(S,i))! \tag{19}$$

where the symbol ! represents the elimination of labels according to the partial ordering relation. In the tests carried out, a reduction in the number of labels occurred at each translation by function f_{ij}.

At the final iteration, $k = 2n + 1$, there is only one state $(\{1, \ldots, 2n + 1\}, 2n + 1)$. All nodes have been visited from the departure point to the arrival point. The minimum value of the objective function is given in the label with the smallest cost, i.e. the last in the list $H(\{1, \ldots, 2n+1\}, 2n+1)$ if this set is non-empty. If the set is empty, the problem is infeasible; this can be detected at any given iteration if no labels are created.

3. STATE ELIMINATION.

As the algorithm is based on dynamic programming, it is useful to limit the number of states created. For a state (S,i) to which node j is to be added, the first four elimination tests described for state ($S \cup \{j\}$,j) are independent of the terminal node i, while the next four tests depend on the labels in H(S,i). To simplify the execution of the tests, information on the states is stored in a three level structure. At the first level, we have the set S and its load y(S). At the second level, for the given set S, we have the terminal nodes $i \in S$ used to form the states (S,i). At the third level, for a state (S,i), the reduced set of labels H(S,i) is stored. (A simple example of this structure is presented in Section 4.1.)

3.1. Elimination criteria based on the set S only. Let S be a set at iteration k-1 to which we wish to add a node j to form state ($S \cup \{j\}$,j) at iteration k. We eliminate this state if one or more of the following criteria is not respected.

#1 : node j must not have been previously visited : $j \in \overline{S}$.

#2 : if j is a destination, then the origin node j-n must have been previously visited (precedence constraint): if $j \in \{n+1,\ldots,2n\}$ then $j-n \in S$.

#3 : if j is an origin, vehicle capacity constraints must be respected: if $j \in \{1,\ldots,n\}$ then $y(S) + C_j \leq C$.

#4 : time $t_j = A_j$ is the earliest time at which node j can be visited. Given this time, it must be possible to visit each subsequent unvisited node $l \in \overline{S \cup \{j\}}$ while respecting the time constraint:

$$A_j + M_j + T_{jl} \leq B_l, \quad \text{for all } l \in \overline{S \cup \{j\}}.$$

For all unvisited nodes, criterion #4 is not tested. The nodes are preordered in decreasing order of their upper bounds; it is reasonable to believe that this order might resemble the optimal order for visiting the nodes. The test is carried out

for the most likely candidate nodes to be visited next: at iteration k+1, only the unvisited nodes ranked from k-2 to k+4 are tested. If one of these nodes cannot be visited after j, then state $(S \cup \{j\}, j)$ is rejected.

3.2. Elimination criteria based on the state (S,i). Let (S,i) be a state at iteration k-1 and t_i be the earliest arrival time at node i given by the first label in H(S,i). State $(S \cup \{j\}, j)$ will be eliminated if it is impossible to create labels. The four criteria stated formally below are based on:

- the time of the visit to node j (#5)
- the time of visit to a node subsequent to node j (#6)
- visits to two destinations whose origins are already visited, following the visit to node j (#7)
- visits to two new origins following node j (#8).

Note that criteria #1, 2, 3 and 5 determine whether a state $(S \cup \{j\}, j)$ is or is not ante-feasible, while criteria #4, 6, 7 and 8 determine whether this state is not post-feasible.

#5 : time constraint must be respected : $t_i + H_i + t_{ij} \leq B_j$.

For the three following criteria, the earliest time at which j can be visited after i is denoted by $t_j = \max [A_j, t_i + H_i + T_{ij}]$.

#6 : if we suppose that node j is visited at time t_j, it must be possible to visit each unvisited node $\ell \in \overline{S \cup \{j\}}$ while respecting the time constraints : $t_j + H_j + T_{j\ell} \leq B_\ell$, for all $\ell \in \overline{S \cup \{j\}}$. This criterion tightens criterion #4 and is tested in the same way.

#7 : if we suppose that node j is visited at time t_j, it must be possible to visit all pairs of destinations $n+\ell_1$ and $n+\ell_2$ whose origins ℓ_1 and ℓ_2 have been previously visited while respecting the time constraints. With two visiting permutations for $n+\ell_1$ and $n+\ell_2$, we must satisfy:

$$\max [A_{n+\ell_1}, t_j+M_j+T_{j,n+\ell_1}] + M_{n+\ell_1} + T_{n+\ell_1,n+\ell_2} \leq B_{n+\ell_2}$$

or

$$\max [A_{n+\ell_2}, t_j+M_j+T_{j,n+\ell_2}] + M_{n+\ell_2} + T_{n+\ell_2,n+\ell_1} \leq B_{n+\ell_1}$$

with $\ell_1, \ell_2 \in S \cup \{j\}$ and $n+\ell_1, n+\ell_2 \in \overline{S \cup \{j\}}$.

Criterion #7 is tested for only one pair $(n+\ell_1, n+\ell_2)$. The destinations are preordered in non-decreasing order of their upper bounds: the first two unvisited destinations whose origins have been visited are chosen.

#8 : if node j is visited at time t_j, it must be possible to visit all pairs of unvisited origins ℓ_1 and ℓ_2 while respecting the time constraints. With two visiting permutations for $n+\ell_1$ and $n+\ell_2$, we must satisfy:

$$\max [A_{\ell_1}, t_j + M_j + T_{j,\ell_1}] + M_{\ell_1} + T_{\ell_1,\ell_2} \leq B_{\ell_2}$$

or

$$\max [A_{\ell_2}, t_j + M_j + T_{j,\ell_2}] + M_{\ell_2} + T_{\ell_2,\ell_1} \leq B_{\ell_1}$$

with $\ell_1, \ell_2 \in \{1,\ldots,n\} \cap S \cup \{j\}$.

Criterion #8 is tested for only one pair (ℓ_1, ℓ_2). The origins are preordered in non-decreasing order of their upper bounds : the two first unvisited origins are chosen.

Note that criterion #6 (and the same applies to criterion #8) cannot be improved by also considering the destination corresponding to the unvisited origin. We would then have to satisfy:

$\max [A_j, t_j + M_j + T_{j\ell}] + M_\ell + T_{\ell,n+\ell} \leq B_{n+\ell}$, $\ell, n+\ell \in S \cup \{j\}$.
But the time intervals are always defined so that $B_{n+\ell} > B_\ell + (M_\ell + T_{\ell,n+\ell})$. This is a condition on the input data. Given this condition, the feasibility check for the destination follows from the check for the origin.

3.3. Elimination criterion in the case with several clients at the same location. Criterion #9, tested after criterion #5, was developed to limit the proliferation of states when i and j represent the same physical location. At a given iteration, the states $(S-\{j\},i)$ and $(S-\{i\},j)$ may exist simultaneously if the constraints are sastisfied. At the following iteration, if both states (S,i) and (S,j) are feasible, only one should be retained because the two will produce solutions of the same cost (as the distance between nodes i and j is zero).

Let i and j be the two nodes at the same location with time windows $[A_i,B_i]$ and $[A_j,B_j]$ and handling times M_i and M_j (travel times are zero). The state (S,i) or (S,j) retained is the one which has the earliest finishing time at that physical location. If t is the arrival time at this location, the state (S,j) is retained (i.e. the order $i \to j$ is imposed) if

$$\text{Max}[A_j,\ \max[A_i,t]+M_i]+M_j < \max[A_i,\ \max[A_j,t]+M_j]+M_i. \quad (20)$$

The left-hand side of (20) corresponds to the order $i \to j$ and conversely for the right-hand side. It can be verified that this inequality is valid if $A_i < A_j$. This condition depends partly on the time t since both states (S,i) and (S,j) have to be feasible in order to make a choice. If $A_i = A_j$, $i \to j$ or $j \to i$ may be chosen arbitrarily. If the vehicle capacity is constraining (which was not the case in our test problems), relation (20) is sufficient to impose the order $i \to j$ in the case where the nodes are two origins or two destinations. When one is an origin and the other is a destination, customers must be dropped off before new ones can be picked up, i.e. node i must be a destination and node j be an origin.

The reader should note that this criterion does not eliminate states generated from $(S-\{j\},i)$ or $(S-\{i\},j)$ and other nodes which are not at the same physical location as i and j. So, if this is optimal, the vehicle may only handle part of the

service at a given physical location at the first visit, and it may then return to serve the rest.

This criterion reduces the number of states generated in the following way. (An example is presented in Figure 1 with nodes i, j and k and $A_i < A_j < A_k$.)

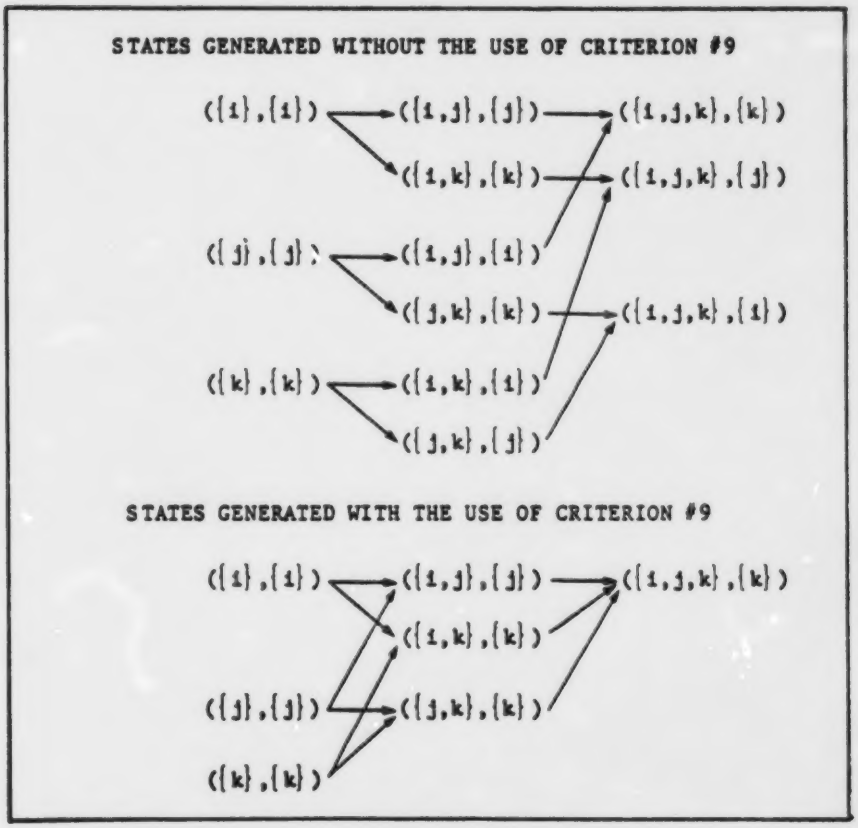


Figure 1 : Effect of Elimination Criterion #9 on the Number of States Generated.

Suppose v nodes are located at the same place. Each time it is possible to visit these v nodes in the algorithm in any

order, the number of states generated during the v consecutive iterations is given by:

$$C_1^v + C_2^v + \ldots + C_v^v = 2^v - 1 \qquad (21)$$

using criterion #9. If criterion #9 is not used, this number becomes

$$vC_1^{v-1} + vC_1^{v-1} + \ldots + vC_{v-1}^{v-1} = v2^{v-1} \qquad (22)$$

Criterion #9 thus produces a reduction by a factor of approximately 2/v in the number of states generated.

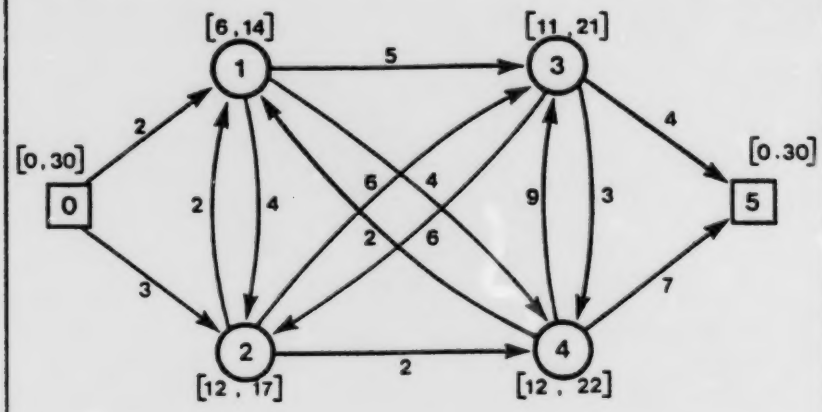
Finally, criterion #9 may be very useful when several nodes are very close to one another without being at exactly the same place. A simple way to limit the proliferation of states is to group the nodes at the same place and apply this criterion. The loss in optimality will not be great, since the distances involved are very small, but computation times may be significantly reduced.

4. NUMERICAL RESULTS.

In this section, we illustrate how the algorithm works, with a simple example, give certain details concerning its implementation, and finally present the results of the tests carried out.

4.1. Example. Figure 2 shows an example with only two requests, where transportation is required from node 1 to node 3, and from node 2 to node 4. The time interval during which each node is to be visited is indicated on the diagram in Figure 2. To simplify the presentation, we assume that the vehicle capacity will not be violated, and that the number associated with each arc represents simultaneously the cost and the travel plus handling times. At the bottom of Figure 2, the three level information structure is illustrated: the sets and terminal nodes from the states, while the (time, cost) labels are at the third level.

At the first iteration, the origins are visited and states



k	STATES			LABEL'S NUMBERS	PRECEDING LABEL NUMBER
	SETS	TERMINAL NODES	LABELS		
1	{1}	1	(6,2)	1	∅
	{2}	2	(12,3)	2	∅
2	{1,2}	1	(14,5)	3	2
		2	(12,6)	4	1
	{1,3}	3	(11,7)	5	1
3	{1,2,3}	2	(17,13)	6	5
		3	(18,12)	7	4
			(19,10)	8	3
4	{1,2,3,4}	4	(19,15)	9	6
			(22,13)	10	8
5	{1,2,3,4,5}	5	(26,22)	11	9
			(29,20)	12	10

Optimal route : 5←4←3←1←2←0 : Cost = 20; Time = 29

Alternative route : 5←4←2←3←1←0 : Cost = 22; Time = 26

Figure 2 : Numerical Example.

($\{1\}$,1) and ($\{2\}$,2) are created with one label each. At iteration k=2, the feasible states are constructed either from one origin and its destination (i.e. state ($\{1,3\}$,3) with a single label), or from two origins (i.e. ($\{1,2\}$,1) and ($\{1,2\}$,2) with one label each. The infeasible states are eliminated as follows: by criterion #1, states ($\{1\}$,1) and ($\{2\}$,2); by criterion #2, states ($\{1,4\}$,4) and ($\{2,3\}$,3); and by criterion #6, state ($\{2,4\}$,4) as it is impossible to visit node 1 afterwards while satisfying the time window constraint. At iteration k=3, state ($\{1,2,3\}$,2) is created with the label (17, 13) as well as state ($\{1,2,3\}$,3) with two labels (18,12) and (19, 10) placed in decreasing time order. State ($\{1,2,4\}$,4) is eliminated by criterion #6 as it is impossible to visit node 3 afterwards while satisfying the time window constraint. At iteration k=4, only one state can be constructed from the states of iteration 3, and adjusting the previous labels with numbers 6, 7 and 8, we obtain only two labels as follows :

$(17,13) \xrightarrow{+2} (19,15)$

$(18,12) \xrightarrow{+3} (21,15)$ eliminated by label (19,15)

$(19,10) \xrightarrow{+3} (22,13)$

Finally, we obtain two routes at iteration k=5 : the optimal route has a cost of 20 and finishes at time 29, while the quickest route takes a time of 26, at a cost of 22. The routes are identified by backtracking the label numbers.

Note that each (time, cost) label corresponds to an ordered visit to a set of nodes. The three level structure proposed uses less space and simplifies the use of the state elimination criteria.

4.2. Implementation. We now describe certain details of the implementation of the numerical tests. The initialization of the algorithm is carried out directly at iteration k=2 by considering all distinct origin pairs $i \neq j$, $1<i$, $j<n$ satisfying criteria

#3 to #9, and all origin-destination pairs i and n+i satisfying criteria #6 to #8. The cost between two nodes is given in minutes. In fact, the only values stored in a non-symmetric matrix are the $T_{ij} + M_i$ (travel time between i and j plus handling time at node i). These same values are used for the costs $D_{ij} = T_{ij} + M_i$, as the vehicle visits all the nodes and the sum of handling times is a constant.

The data structure with three levels is retained in a packed form. One word per set contains the first level information: the feasible set, and a pointer to the first terminal node. One word per state contains the second level information: terminal node, load, pointer to the following terminal node for this same set and a pointer to the first label. One word per label contains third level information: time, cost, pointer to the state (S,i), and two pointers to the preceding and following labels in the list H(S,i).

To limit memory space requirements, at iteration k, we store at the first and second levels only the feasible sets and terminal nodes of the current iteration k and the preceding iteration k-1. Furthermore, at the end of iteration k, the labels of iteration k-1 which were not used to generate new labels are eliminated from the structure.

4.3. Tests. The 98 problems investigated had from 5 to 40 requests and were derived from the cities of Montréal, Toronto and Sherbrooke in Canada. The problems with over 25 requests were constructed by combining smaller problems based on morning, afternoon, and evening routes. The main characteristics of these problems are presented in Table 1. The time windows associated with the origins and destinations are of the order of 40-50 minutes, with a minimum of 30 minutes and a maximum of 82 minutes. The "route diameter" is a measure of the size of the area covered by a route. It is expressed in terms of the travel time along a straight line joining the two most distant points

Table 1: Problem characteristics.

CITY	NUMBER OF PROBLEMS	TIME WINDOWS[1]			ROUTE DIAMETER[2]		MEAN ARC SIZE[2]
		MIN	MEAN	MAX	MEAN	MAX	
MONTREAL	65	30	38	51	44	70	12
SHERBROOKE	9	30	38	59	24	33	9
TORONTO	24	40	53	82	43	63	13

[1] In seconds.
[2] Travel time in minutes.

visited by the route. This diameter is on average 43-44 minutes for the large cities (Montreal and Toronto), and around 24 minutes for Sherbrooke, a small town. The last column gives the average travel time between two consecutive points along the optimal routes. Note that the time windows are large compared with the mean arc size of around 10 minutes. The problems tested therefore may have a very large number of feasible routes; each point has on average 6 possible successors respecting the time windows. In addition, vehicle capacity was always sufficient and all problems were feasible.

Figure 3 shows the execution time in seconds as a function of problem size. Test problems were solved on the University of Montreal Cyber 173 computer using the FORTRAN 5 (opt=2) compiler. Almost all (76 out of 82) of the original problems with $n \leq 25$ were solved in less than two seconds. Over the whole set of problems, the relationship between execution time and problem size is almost linear. This relationship can be expected as there is no interaction between two requests separated by several hours. Tests on additional larger scale randomly generated feasible problems produced similar results.

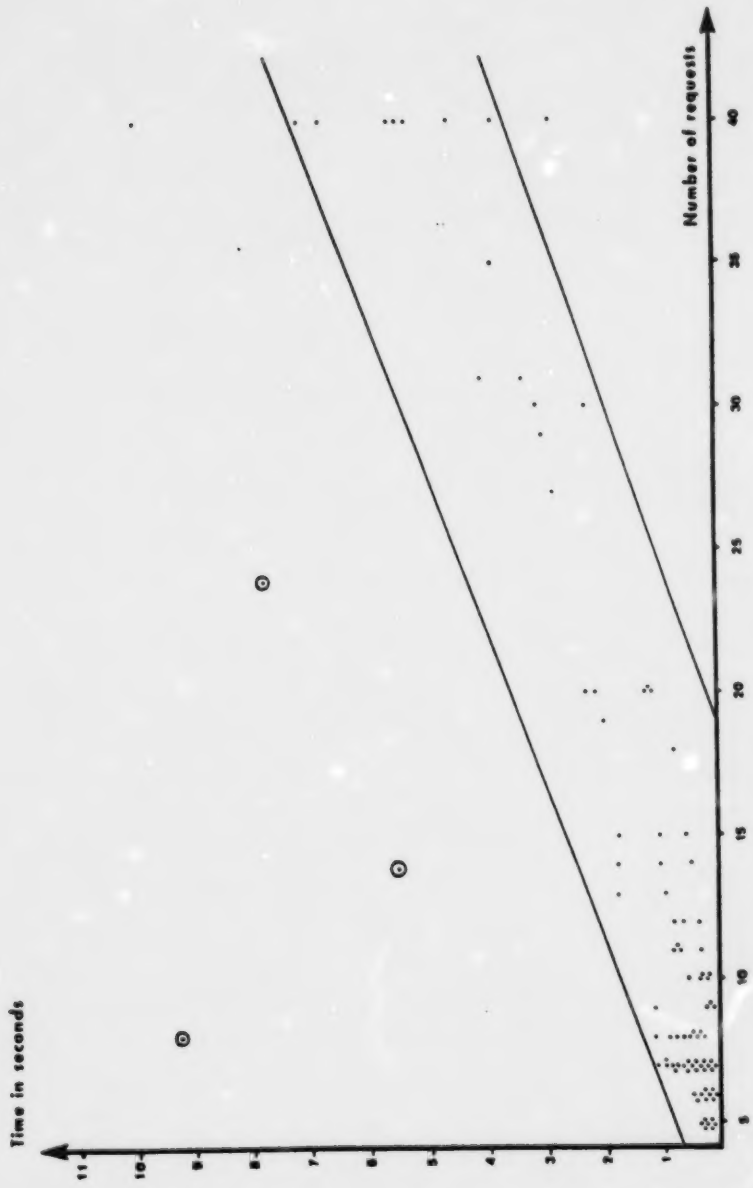


Figure 3: Execution Time as a Function of the Number of Requests

For the more detailed study of the numerical results, we excluded three problems which are easily identifiable in Figure 3 as having high execution times (with n=8, 14 and 24). These problems have origins and/or destinations which are close together for several requests, with very similar time windows. For example, for the excluded problem with n=8, each node has 11 possible successors on average while respecting the time constraints. This results in a proliferation of states as occurs in the treatment of the traveling salesman problem by dynamic programming.

Table 2 presents various averages by class and size of problem. We noted above the linear relationship between execution time and problem size. The graphical representation of the number of states generated also shows the same relationship. The number of terminal nodes per feasible set is relatively small, less than two on average. The label elimination process using the partial ordering relation reduces the number of labels to be stored for each state to two on average. Even if the elimination is not very strong, it avoids the generation of all successors of these labels and it produces a great effect. The proportional relationship between the number of states generated and the problem size n indicates that the average number of states per iteration is almost constant. Dividing by 2n iterations, we obtain on average five to seven states per iteration. The memory used is thus small. Without including the matrix structure of the values $(T_{ij} + M_i)$, 97 of the 98 problems handled were solved with less than 1500 words, the exception being the difficult problem of size n=8 with 2661 words.

Table 3 shows elimination statistics on the criteria described in Section 3. Criteria #1 and #2 (new node and precedence), which are not really tests but just part of the definition of the successor states, eliminate over 60% of the states in general. The effect of other elimination criteria has been

Table 2: Summary of Results of 95 Problems.

SIZE n	NUMBER OF PROBLEMS	STATES[1]	TERMINAL NODES[2]	LABELS[3]	EXECUTION TIME[4]
5-7	36	57	1.54	1.14	0.31
8-10	18	86	1.59	1.24	0.47
11-19	19	194	1.70	1.37	1.23
20-30	10	287	1.75	1.25	2.57
31-40	12	431	1.72	1.28	5.27

1 Average total number of remaining states per problem.
2 Average number of terminal nodes per feasible set.
3 Average number of labels per state.
4 Average execution time in seconds (CYBER 173).

presented in the order of their application (4, 5, 9, 6, 7, 8). Each column indicates the percentage of states tested with the criterion which have been eliminated (example: for size 31-40, the average total number of states was 23869 + 431 = 24300, criteria #1 and #2 eliminated 17010 states; the value 70% is calculated by the ratio 17010/2400, the 7290 remaining states were tested by criterion #4 and 6342 were eliminated giving 87% success based on the ratio 6342/7290, ...).

Table 3: Elimination statistics by criterion.

SIZE n	STATES[1] ELIMINATED PER PROBLEM	#1 & #2 (2)	#4 (2)	#5 (2)	#9 (2)	#6 (2)	#7 (2)	#8 (2)
5-7	470	65	22	2	9	43	5	11
8-10	910	65	45	5	7	47	8	5
11-19	3058	69	63	6	7	41	7	5
20-30	6936	69	70	13	13	49	8	7
31-40	23869	70	87	18	18	45	6	3

1 Average number of states eliminated by problem.
2 Percentage of states tested with this criterion which have been eliminated.

Criterion #4 (visit to one additional node if node j is visited at time A_j) which test "post-feasibility" is the most important and the performance increases with problem size (from 22% to 87%).

The performance of criterion #5 (time window constraint) also increases with problem size (from 2% to 18%). The percentage of states eliminated by criterion #9 (same location) increases with problem size (from 9% to 18%) and is dependent on the percentage of multiple stops at the same physical location in the test problems. Criterion #6 (visit to one additional node if node j is visited at time t_j) is very important for every problem size (from 40% to 50%). The performance of criteria #7 and #8 (visit to two additional nodes) is less important while criterion #3 (capacity) is inactive.

5. CONCLUSION.

Our algorithm easily solves problems with 40 requests. The corresponding traveling salesman problem has 80 nodes and could certainly not be solved by dynamic programming (see Bellmore and Nemhauser (1983)). The effectiveness of our dynamic programming algorithm is largely due to the use of efficient elimination criteria for states which are infeasible because of the additional constraints on the route. Problems with over 40 requests were not tested because, in the context of transportation for the handicapped, a vehicle rarely handles more than around twenty requests in a day. As the number of states and the computation times are small and as these increase linearly with the number of requests, and as the memory required is very small and does not increase with the number of requests, much larger problems could in fact be solved.

This algorithm is currently being used for the optimization of routes produced by a heuristic for the problem with several vehicles. The distance traveled was reduced by 4% to 5% without diminishing the quality of the service. The algorithm could also

be used to verify the feasibility of serving a set of requests with one vehicle in a "cluster first-route second" approach.

ACKNOWLEDGMENTS

The authors wish to thank the Federal Government of Canada (NSERC and PUT programs), the Quebec Government (FCAR program), and L'École des Hautes Études Commerciales de Montréal, for their financial support.

REFERENCES

Armstrong, G.R. and Garfinkel, R.S. (1982). Dynamic programming solution of the single and multiple pick-up and delivery problem with application to dial-a-ride. Working Paper #162, College of Business Administration, University of Tennessee, Knoxville, TN.

Bélisle, J.-P., Soumis, F., Roy, S., Desrosiers, J., Dumas, Y. and Rousseau, J.-M. (1984). The impact on vehicle routing of various operational rules of a transportation System for handicapped persons. Publication #363*, Centre de recherche sur les transports, Université de Montréal, Montréal, Canada H3J 3J7. (Presented at the Third International Conference on Mobility and Transport of Elderly and Handicapped Persons, Orlando, Florida).

Bellmore, M. and Nemhauser, G.L. (1968). The travelling salesman problem - a survey. Operations Research 16, 538-558.

Bodin, L., Golden, B., Assad, A. and Ball, M. (1983). Routing and scheduling of vehicles and crews - The state of the art, Computers and Operations Research 10, 63-211.

Bodin, L. and Sexton, T. (1986). The multi-vehicle subscriber dial-a-ride problem. The Delivery of Urban Services, TIMS Studies in the Management Science 22, forthcoming.

Desrosiers, J., Dumas, Y. and Soumis, F. (1984). The multiple vehicle many to many routing problem with time windows. Publication #362, Centre de recherche sur les transports, Université de Montréal, Montréal, Canada H3C 3J7.

Desrosiers, J.. Pelletier, P. and Soumis, F. (1983). Plus court chemin avec contraintes d'horaires, R.A.I.R.O. 17, 4, 357-377.

Jaw, J., Odoni, A., Psaraftis, H. and Wilson, N. (1984). A heuristic algorithm for the multi-vehicle advanced request dial-a-ride problem with time windows. Working Paper MIT-UMTA-83-L, MIT, Cambridge, Ma 02139.

Psaraftis, H. (1980), A dynamic programming solution to the single vehicle many to many immediate request dial-a-ride problem. Transportation Science 14 (2), 130-154.

Psaraftis, H. (1982). An exact algorithm for the single vehicle many to many dial-a-ride problem with time windows, Technical Note OE-UMTA-82-2, M.I.T., Cambridge, MA 02139.

Psaraftis, H. (1983). Analysis of an $O(N^2)$ heuristic for the single vehicle many-to-many euclidian dial-a-ride problem, Transportation Research B 17B (2), 133-145.

Roy, S., Rousseau, J.-M., Lapalme, G. and Ferland, J.A. (1984a). Routing and scheduling for the transportation of disabled persons - The algorithm. TP 5596E. Transport Development Centre, Transport Canada Montréal, Canada H2Z 1X4.

Roy, S., Rousseau, J.-M., Lapalme, G. and Ferland, J.A. (1984b), Routing and scheduling for the transportation for disabled persons - The tests. TP 5598E. Transport Development Centre, Transport Canada, Montréal, Canada H2Z 1X4.

Sexton, T. and Bodin, L. (1985a), Optimizing single vehicle many-to many operations with desired delivery times: I. Scheduling. Transportation Science 10 (4), 378-410.

Sexton, T. and Bodin, L. (1985b), Optimizing single vehicle many-to many operations with desired delivery times: II. Routing. Transportation Science 10 (4), 411-435.

Received 11/84; Revised 6/28/86.

4

Optimisation de l'horaire d'un itinéraire

Ce chapitre présente un algorithme exact pour la troisième phase de notre approche, soit celle de l'optimisation de l'horaire des itinéraires. Ce chapitre fera l'objet d'un article. Il se trouve donc écrit sous la forme d'un article. La notation est un peu différente mais ne devrait pas causer d'ambiguïté. La contribution majeure de ce chapitre est le développement d'un algorithme dual, de complexité linéaire, qui permet de considérer des intervalles de temps aux arrêts pour l'heure de visite et des fonctions d'inconvénient convexes sur cette heure de visite.

4.1. Introduction

Soit une suite de noeuds définissant l'itinéraire d'un véhicule. L'heure de service à chacun des noeuds est restreinte par un intervalle de temps. Il existe généralement plusieurs horaires réalisables pour ce véhicule; plusieurs heures de service aux noeuds peuvent être réalisables et le temps d'attente du véhicule peut être réparti de plusieurs façons au cours de l'itinéraire. Le problème considéré dans cet article consiste à choisir parmi ces horaires réalisables celui qui minimise la somme des fonctions d'inconvénients associées aux heures de service aux noeuds. L'inconvénient à un noeud est modélisé par une fonction convexe de l'heure de service.

Lorsque les inconvénients sont tels que les meilleures heures de service aux noeuds sont toutes au plus tôt ou toutes au plus tard, alors le problème est trivial et aucun algorithme spécial n'est nécessaire. Autrement, il s'agit d'un problème d'optimisation de plusieurs variables avec contraintes: les heures de service doivent respecter les contraintes d'intervalles de temps et elles sont liées par le temps de déplacement du véhicule.

Solomon et Desrosiers (1988) et Desrochers et al. (1988) présentent des revues de la littérature sur les problèmes d'optimisation d'itinéraires et d'horaires de véhicules avec contraintes d'horaires. Le problème traité dans cet article intervient comme une étape dans plusieurs approches de résolution pour ces problèmes d'optimisation. Ainsi dans l'approche traitant simultanément les itinéraires de tous les véhicules, les étapes sont les suivantes:

1) Optimisation simultanée des itinéraires de tous les véhicules;
2) Optimisation de l'horaire de chaque véhicule.

La première étape minimise les coûts de déplacement tout en visitant chaque noeud dans son intervalle de temps. Les intervalles de temps sont généralement suffisamment petits pour assurer un minimum de qualité de service durant l'optimisation des coûts. Des algorithmes optimaux pour cette étape ont été développés en transport des marchandises (Dumas, 1985 et Kolen et al., 1987), en transport scolaire (Desrosiers et al., 1984), en transport urbain (Soumis et al., 1985) et en transport des personnes handicapées (Desrosiers et al., 1988). La seconde étape, qui est traitée dans cet article, permet l'optimisation finale de l'horaire de chacun des véhicules en considérant les inconvénients par rapport aux heures de service désirées à chaque noeud.

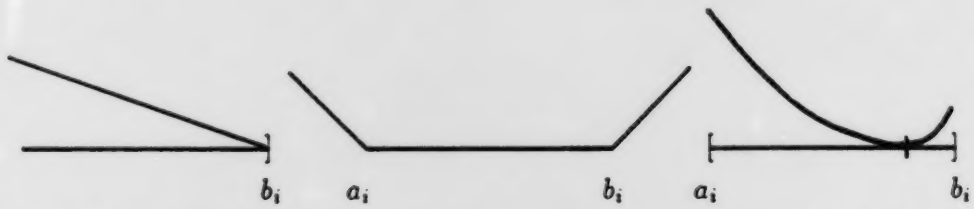
Ce problème apparaît également à la toute fin de l'approche séparant le problème par véhicule et dont les étapes sont les suivantes:

1) Regroupement des noeuds;
2) Optimisation de l'itinéraire de chaque véhicule;
3) Optimisation de l'horaire de chaque véhicule.

Dans ce cas, après le regroupement des noeuds par véhicule à la première étape, l'optimisation des coûts de déplacement se fait véhicule par véhicule à la deuxième étape. Pour le regroupement des noeuds, plusieurs méthodes approximatives ont été élaborées selon le contexte d'application (voir Bodin et al., 1983) alors que des algorithmes optimaux pour la seconde étape ont été développés en transport avec cuillettes et livraisons (Desrosiers et al., 1986a et Psaraftis, 1983).

Deux cas particuliers du problème de l'optimisation de l'horaire d'un véhicule ont été traités. Sexton et Bodin (1985a, 1985b) et Sexton et Choi (1986) utilisent l'approche séparant le problème par véhicule. Ils résolvent simultanément les étapes 2) et 3) précédentes à l'aide d'une décomposition de Benders. Le problème maître détermine l'itinéraire du véhicule et le sous-problème détermine l'horaire. Les approches se distinguent au niveau du problème de l'optimisation de l'horaire d'un véhicule. Sexton et Bodin (1985a) considèrent uniquement une heure maximale de service b_i à un noeud i plutôt qu'un intervalle de temps et

l'inconvénient est modélisé par une fonction linéaire (figure 1a). Sexton et Choi (1985) ne considèrent aucune limite sur les heures de service aux noeuds mais ils modélisent l'inconvénient par une fonction linéaire en trois morceaux pour favoriser les heures de l'intervalle $[a_i, b_i]$ (figure 1b). Ces deux articles proposent des algorithmes ayant des temps de calcul de l'ordre du nombre de noeuds. Nous proposons un modèle plus général (figure 1c) où l'inconvénient est modélisé par une fonction convexe définie sur l'intervalle de temps $[a_i, b_i]$ et démontrons de façon plus concise un résultat du même ordre de complexité.



a) Sexton et Bodin b) Sexton et Choi c) modèle proposé

Figure 1: Intervalle de temps et fonction d'inconvénient associés à chaque noeud.

4.2. Formulation mathématique

Soient $\mathcal{P}$ le chemin visitant, dans l'ordre, les noeuds $1, 2, \ldots, n$ et $N = \{1, 2, \ldots, n\}$ l'ensemble de ces noeuds. Nous représentons par $t_{i,i+1}$, $i = 1, \ldots, n-1$, le temps nécessaire pour aller du noeud i au noeud $i+1$. Pour chaque noeud $i \in N$, la variable T_i représente l'heure de service et la fonction convexe d'inconvénient $f_i(T_i)$ est définie sur l'intervalle de temps $[a_i, b_i]$.

Nous formulons le problème (P) de l'optimisation de l'horaire de la façon suivante:

$$(P) \qquad \min \sum_{i \in N} f_i(T_i) \qquad (1)$$

sujet à

$$T_i + t_{i,i+1} \leq T_{i+1} \qquad i = 1, 2, \ldots, n-1 \qquad (2)$$

$$a_i \leq T_i \leq b_i \qquad i = 1, 2, \ldots, n. \qquad (3)$$

L'objectif (1) consiste à minimiser la somme des fonctions d'inconvénients sur l'heure de service à chaque noeud, sujet aux contraintes de liaison (2) qui décrivent la compatibilité de l'horaire avec le chemin et les contraintes d'intervalles (3) qui imposent les intervalles de temps aux noeuds.

Nous utilisons, par la suite, la notation $(\bar{P})$ pour se référer au problème (P) sans les contraintes de liaison (2).

4.3. Pré-traitement

Le pré-traitement consiste à modifier les données du problème afin d'obtenir un problème équivalent respectant les propriétés (4) et (5) ci-dessous:

$$a_i + t_{i,i+1} \leq a_{i+1} \qquad i = 1, 2, \ldots, n-1 \qquad (4)$$

$$b_i + t_{i,i+1} \leq b_{i+1} \qquad i = 1, 2, \ldots, n-1. \qquad (5)$$

En vue de satisfaire ces propriétés, on modifie les données récursivement, dans l'ordre indiqué, à l'aide des deux relations suivantes:

$$a_{i+1} := \max \{a_{i+1},\, a_i + t_{i,i+1}\} \quad \text{pour } i = 1, 2, \ldots, n-1 \qquad (6)$$

$$b_i := \min \{b_i,\, b_{i+1} - t_{i,i+1}\} \quad \text{pour } i = n-1, n-2, \ldots, 1. \qquad (7)$$

Les relations (6) et (7) réduisent les intervalles de temps aux noeuds, le long du chemin, pour éliminer les parties inutiles. Notons que la complexité de ce pré-traitement est de l'ordre du nombre de noeuds n. Le problème est irréalisable si après le pré-traitement il existe un noeud i tel que $a_i > b_i$. Pour la suite, nous supposerons que les données ont été pré-traitées et que le problème est réalisable.

4.4. Réduction du problème

Dans l'approche de résolution décrite à la section suivante, les contraintes de liaison (2) sont d'abord relaxées, puis réintroduites au besoin sous forme d'égalités. Pour $n \geq 2$, soit $C \subset \{1,2,\ldots,n-1\}$ l'ensemble des contraintes d'égalité. On formule le problème (P_C) comme suit:

$$(P_C) \qquad \min \sum_{i \in N} f_i(T_i) \qquad (1)$$

sujet à

$$T_i + t_{i,i+1} = T_{i+1} \qquad i \in C \quad (8)$$

$$T_i + t_{i,i+1} \leq T_{i+1} \qquad i \in \{1,2,\ldots,n-1\} - C \quad (9)$$

$$a_i \leq T_i \leq b_i \qquad i = 1,2,\ldots,n. \quad (3)$$

Ce problème est identique au problème original (P) si $C = \emptyset$. Pour $C \neq \emptyset$ et $i \in C$, T_i est lié à T_{i+1} par la contrainte $T_i + t_{i,i+1} = T_{i+1}$; on peut alors effectuer un changement de variable et éliminer la variable T_i. En éliminant ainsi toutes les variables T_i pour $i \in C$ et en modifiant légèrement la notation on obtient une formulation plus compacte avec $n - |C|$ variables.

Introduisons maintenant la notation pour écrire cette formulation. Soit I_C l'ensemble des couples de noeuds adjacents du sous-chemin du chemin $\mathcal{P}$ comprenant les noeuds de $N - C$. Par exemple, $I_\emptyset = \{(1,2),(2,3),\ldots,(n-1,n)\}$ et $I_{\{k\}} = \{(1,2),(2,3),\ldots,(k-2,k-1),(k-1,k+1),(k+1,k+2),\ldots,(n-1,n)\}$. Nous représentons par $t_{i,j}$, $(i,j) \in I_C$, le temps nécessaire pour aller du noeud i au noeud j en suivant le chemin $\mathcal{P}$. Les fonctions f_i et les constantes b_i sont indicées par C pour indiquer une différence possible par rapport aux définitions initiales. La formulation plus compacte (Q_C) obtenue est la suivante:

$$(Q_C) \qquad \min \sum_{i \in N-C} f_i^C(T_i) \qquad (10)$$

sujet à

$$T_i + t_{i,j} \leq T_j \qquad (i,j) \in I_C \qquad (11)$$

$$a_i \leq T_i \leq b_i^C \qquad i \in N - C \qquad (12)$$

Les nouveaux temps de déplacement sont calculés comme suit:

$$t_{i,j} := \sum_{k=i}^{j-1} t_{k,k+1} \qquad (i,j) \in I_C \qquad (13)$$

alors que pour la fonction f_i^C et la constante b_i^C, il faut d'abord définir l'indice ℓ_i. Pour $i \in N - C$ tel que $i - 1 \in C$, soit ℓ_i tel que $\ell_i - 1 \notin C$ et $\{\ell_i, \ell_i + 1, \ldots, i-1\} \subseteq C$ (c'est-à-dire que l'arc $(\ell_i - 1, i)$ appartient à I_C si $\ell_i > 1$). Alors pour $i \in N - C$, f_i^C et b_i^C sont définies comme suit:

$$f_i^C(T_i) := \begin{cases} f_i(T_i) & \text{si } i-1 \notin C \\ \sum_{k=\ell_i}^{i} f_k(T_i - t_{k,i}) & \text{sinon;} \end{cases} \qquad (14)$$

$$b_i^C := \begin{cases} b_i & \text{si } i-1 \notin C \\ b_{\ell_i} + t_{\ell_i,i} & \text{sinon.} \end{cases} \qquad (15)$$

Il est intéressant de noter les propriétés suivantes:

Propriété 1: Les fonctions f_i^C sont convexes comme somme des fonctions convexes f_i.

Propriété 2: Les intervalles $[a_i, b_i^C]$, $i \in N - C$ satisfont les propriétés découlant du pré-traitement.

Propriété 3: Le problème (Q_C) prend la même forme que le problème original (P) mais avec seulement $n - |C|$ variables.

Propriété 4: Il existe une bijection entre les solutions des problèmes (P_C) et (Q_C). Ainsi, on obtient facilement une solution optimale du problème (P_C) à

partir d'une solution optimale du problème (Q_C) en posant, pour $i = n, n-1, \ldots, 1$

$$T_i := \begin{cases} T_i & \text{si } i \notin C \\ T_{i+1} - t_{i,i+1} & \text{sinon.} \end{cases} \tag{16}$$

Dans la suite de l'article, on note par $(\bar{Q}_C)$ le problème (Q_C) pour lequel on a relaxé les contraintes de liaison (11). Le problème relaxé $(\bar{Q}_C)$ se résout beaucoup plus facilement puisqu'il se décompose en une suite de problèmes unidimentionnels convexes.

4.5. Approche de résolution

L'approche proposée est duale. Elle consiste, à partir du problème relaxé $(\bar{P})$, à réintroduire les contraintes violées une à une jusqu'à ce que toutes les contraintes soit respectées. La structure du problème fait qu'il est possible de réintroduire en même temps plusieurs des contraintes violées avec les inégalités remplacées par des égalités. Toutefois, en introduisant une contrainte à égalité à chaque itération, notre approche génère les solutions optimales de la suite des problèmes relaxés

$$(\bar{Q}_{C^0}), (\bar{Q}_{C^1}), \ldots, (\bar{Q}_{C^r}) \qquad r \leq n-1$$

où l'indice r représente le nombre de contraintes à égalité dans le problème original, avec $C^1 \subset C^2 \subset \cdots \subset C^r$. Le premier problème $(\bar{Q}_{C^0})$ correspond à $(\bar{P})$ alors que le dernier problème $(\bar{Q}_{C^r})$ admet une solution optimale qui satisfait également toutes les contraintes relaxées du problème (Q_{C^r}). Le problème (Q_{C^r}) étant une formulation compacte du problème équivalent (P_{C^r}), on déduit facilement une solution optimale de ce dernier problème (propriété 4). Les résultats suivants montrent que cette solution optimale du problème (P_{C^r}) est également optimale pour le problème original (P).

Lemme 1: Soit le problème (Q_C) avec $0 \leq |C| \leq n-2$, et $f_i^C(T_i)$ convexes sur l'intervalle $[a_i, b_i^C]$, pour $i \in N - C$. Soit $\bar{T} = (\bar{T}_i \mid i \in N - C)$, une

solution optimale du problème relaxé $(\bar{Q}_C)$. Si $\bar{T}_k + t_{k,\ell} \geq \bar{T}_\ell$, $(k,\ell) \in I_C$ alors il existe une solution optimale $T^* = (T_i^* \mid i \in N - C)$ au problème (Q_C) tel que $T_k^* + t_{k,\ell} = T_\ell^*$.

Preuve: Soit $(k,\ell) \in I_C$ tel que $\bar{T}_k + t_{k,\ell} \geq \bar{T}_\ell$ et soit $T^* = (\ldots, T_k^*, T_\ell^*, \ldots)$ une solution optimale du problème (Q_C) tel que $T_\ell^* - T_k^*$ est minimum. Puisque T^* satisfait les contraintes (11) alors $T_\ell^* - T_k^* \geq t_{k,\ell}$. Supposons $T_\ell^* - T_k^* > t_{k,\ell}$ et montrons, par contradiction, que $T_\ell^* - T_k^* = t_{k,\ell}$.

(1) $\bar{T}_k \leq T_k^*$:

Autrement, si $\bar{T}_k > T_k^*$ alors $T' = (\ldots, T_k^* + \epsilon_1, T_\ell^*, \ldots)$, obtenue de T^* en remplaçant T_k^* par $T_k^* + \epsilon_1$ où $\epsilon_1 = \min(T_\ell^* - t_{k,\ell} - T_k^*, \bar{T}_k - T_k^*) > 0$, est réalisable (i.e. $T_k^* + \epsilon_1 + t_{k,\ell} \leq T_\ell^*$), optimal (i.e. $f_k^C(\bar{T}_k) \leq f_k^C(T_k^* + \epsilon_1) \leq f_k^C(T_k^*)$ par définition de $\bar{T}_k$ et la convexité de f_k^C) et contredit le fait que $T_\ell^* - T_k^*$ est minimum (i.e. $T_\ell^* - T_k^* > T_\ell^* - (T_k^* + \epsilon_1)$).

(2) $\bar{T}_\ell \geq T_\ell^*$:

Autrement, si $\bar{T}_\ell < T_\ell^*$ alors $T' = (\ldots, T_k^*, T_\ell^* - \epsilon_2, \ldots)$, obtenue de T^* en remplaçant T_ℓ^* par $T_\ell^* - \epsilon_2$ où $\epsilon_2 = \min(T_\ell^* - t_{k,\ell} - T_k^*, T_\ell^* - \bar{T}_\ell) > 0$, est réalisable (i.e. $T_k^* + t_{k,\ell} \leq T_\ell^* - \epsilon_2$), optimal (i.e. $f_\ell^C(\bar{T}_\ell) \leq f_\ell^C(T_\ell^* - \epsilon_2) \leq f_\ell^C(T_\ell^*)$ par définition de $\bar{T}_\ell$ et la convexité de f_ℓ^C) et contredit le fait que $T_\ell^* - T_k^*$ est minimum (i.e. $T_\ell^* - T_k^* > T_\ell^* - \epsilon_2 - T_k^*$).

Combinant (1), (2) et $T_\ell^* - T_k^* > t_{k,\ell}$ on obtient $\bar{T}_\ell - \bar{T}_k \geq T_\ell^* - T_k^* > t_{k,\ell}$ ce qui contredit $\bar{T}_k + t_{k,\ell} \geq \bar{T}_\ell$. Donc $T_\ell^* - T_k^* = t_{k,\ell}$ et le lemme est démontré. □

Le lemme 1 s'applique directement au problème original (P). Ainsi, si la solution du problème relaxé $(\bar{P}) = (\bar{Q}_{C^0})$ ne satisfait pas toutes les contraintes de (P), on pose $C^1 = \{k\}$, où $k \in \{1, 2, \ldots, n-1\}$. Du lemme, on obtient qu'une solution optimale du problème (P_{C^1}) est également optimale pour le problème original (P). Or, la solution du problème (P_{C^1}) peut être déduite (propriété 4) de la solution du problème réduit (Q_{C^1}) qui conserve les propriétés de convexité des fonctions et du pré-traitement. Par induction sur le nombre de contraintes

à égalité, on montre ainsi qu'une solution optimale du problème (P_{C^r}) avec $C^r = C^{r-1} \cup \{k\}$, où $k \in \{1,2,\ldots,n-1\} - C^r$, est également optimale pour le problème original (P). Puisqu'il existe une bijection entre les solutions des problèmes (P_{C^r}) et (Q_{C^r}), nous obtenons le théorème suivant.

Théorème 1: Soit (Q_{C^r}), $r \leq n-1$, le problème réduit obtenu du problème (P) par transformations successives suite à l'introduction de contraintes à égalité. Soit $\bar{T} = (\bar{T}_i \mid i \in N - C^r)$, une solution optimale du problème relaxé $(\bar{Q}_{C^r})$ satisfaisant toutes les contraintes de (Q_{C^r}); alors pour $i = n, n-1, \ldots, 1$

$$T_i^* := \begin{cases} \bar{T}_i & \text{si } i \in N - C^r \\ T_i^* - t_{i,i+1} & \text{sinon} \end{cases} \tag{17}$$

est une solution optimale du problème original (P).

Afin de compléter la théorie permettant d'élaborer un algorithme efficace résolvant le problème (P), nous démontrons le lemme suivant.

Lemme 2: Soit $0 \leq |C| \leq n-2$ et $\bar{T} = (\bar{T}_i \mid i \in N-C)$ une solution optimale du problème relaxé $(\bar{Q}_C)$ avec $\bar{T}_k + t_{k,\ell} > \bar{T}_\ell$, $(k,\ell) \in I_C$. L'obtention d'une solution optimale du problème $(\bar{Q}_{C\cup\{k\}})$ ne nécessite qu'une seule minimisation unidimentionnelle.

Preuve : Puisque les problèmes relaxés $(\bar{Q}_C)$ et $(\bar{Q}_{C\cup\{k\}})$ ne possèdent aucune contrainte liante entre les variables, ils se décomposent respectivement en $n-|C|$ et $n-|C|-1$ sous-problèmes unidimentionnels indépendants dont $n-|C|-2$ sont identiques entre les deux problèmes. On obtient alors la solution du problème $(\bar{Q}_{C\cup\{k\}})$ à partir de celle du problème $(\bar{Q}_C)$ en gardant la même solution pour les variables $i \in N - C - \{k,\ell\}$ et en résolvant seulement le sous-problème unidimentionnel associé à T_ℓ, soit

$$\min_{a_\ell \leq T_\ell \leq b_k^C - t_{k,\ell}} f_\ell^C(T_\ell) + f_k^C(T_\ell - t_{k,\ell}). \qquad \square$$

4.6. Algorithme

Nous présentons, dans cette section, l'algorithme pour obtenir une solution optimale du problème (P), un exemple de son fonctionnement et un théorème qui énonce que la complexité de l'algorithme, en termes de minimisations unidimentionnelles, est de l'ordre du nombre de noeuds.

L'algorithme est le suivant:

Etape 0 : (Initialisation)

- Pré-traiter les données,
- Soit $\bar{T} = \{\bar{T}_1, \ldots, \bar{T}_n\}$ une solution optimale du problème relaxé $(\bar{P})$,
- Initialiser $I_\emptyset := \{(1,2),(2,3),\ldots,(n-1,n)\}$,
- Poser $C := \emptyset$ et $\ell := 2$.

Etape 1 : (Test de faisabilité sur l'arc (k, ℓ), où $(k, \ell) \in I_C$)

- S'il existe k tel que $(k,\ell) \in I_C$ et $\bar{T}_k + t_{k,\ell} > \bar{T}_\ell$
  alors aller à l'étape 3
  sinon aller à l'étape 2.

Etape 2 : (Avancement d'un noeud: toutes les contraintes avant le noeud ℓ sont satisfaites)

- $\ell := \ell + 1$,
- Si $\ell > n$ alors aller à l'étape 5
  sinon aller à l'étape 1.

Etape 3 : (Construction du problème réduit $(Q_{C\cup\{k\}})$)

Mise à jour de l'énoncé du problème:

- $f_\ell^{C\cup\{k\}}(T_\ell) = f_\ell^C(T_\ell) + f_k^C(T_\ell - t_{k,\ell})$,

- $b_\ell^{C\cup\{k\}} = b_k^C + t_{k,\ell}$,
- s'il existe j tel que $(j,k) \in I_C$
  alors $I_{C\cup\{k\}} = I_C \cup \{(j,\ell)\} - \{(j,k),(k,\ell)\}$,
  $t_{j,\ell} = t_{j,k} + t_{k,\ell}$
  sinon $I_{C\cup k} = I_C - \{k,\ell\}$.

Etape 4 : (Solution optimale du problème $(\bar{Q}_{C\cup\{k\}})$)

- Soit $\bar{T}_\ell$ solution optimale du problème :

$$\min_{a_\ell \le x \le b_\ell^{C\cup\{k\}}} f_\ell^{C\cup\{k\}}(x),$$

- $C := C \cup \{k\}$,
- Aller à l'étape 1.

Etape 5 : (Solution optimale de (P))

- Pour $i = n,\ldots,1$:

$$T_i^* = \begin{cases} \bar{T}_i & \text{si } i \in N - C \\ T_{i+1}^* - t_{i,i+1} & \text{si } i \in C. \end{cases}$$

L'algorithme procède vers l'avant pour satisfaire les contraintes de liaison (2). En tout temps au cours de l'algorithme, les contraintes (2), pour $i < \ell$, sont satisfaites. En effet, à chaque itération seulement la variable T_ℓ change de valeur.

L'étape 0 effectue le pré-traitement des données et le calcul de la solution initiale du problème relaxé. L'étape 1 vérifie si la contrainte associée à l'arc (k,ℓ) est violée (i.e. si $\bar{T}_k + t_{k,\ell} > \bar{T}_\ell$). Lorsque la contrainte n'est pas violée, l'étape 2 augmente ℓ pour passer à la prochaine contrainte. Lorsque la contrainte est violée le problème est réduit à l'étape 3. L'étape 4 procède à la mise à jour de la solution en réévaluant seulement $\bar{T}_\ell$ (lemme 2). Finalement, la dernière

étape de l'algorithme procède au calcul des valeurs des variables éliminées par les réductions. L'ensemble C contient l'ensemble des variables éliminées au cours de l'algorithme.

Exemple: Soit l'énoncé suivant: $N = \{1,2,3,4\}$, $t_{1,2} = t_{2,3} = t_{3,4} = 10$, $[a_1, b_1^\emptyset] = [10,15]$, $[a_2, b_2^\emptyset] = [20,25]$, $[a_3, b_3^\emptyset] = [30,35]$, $[a_4, b_4^\emptyset] = [40,45]$, $f_1^\emptyset(T_1) = 5\cdot(T_1-11)^2$, $f_2^\emptyset(T_2) = 4\cdot(T_2-24)^2$, $f_3^\emptyset(T_3) = 3\cdot(T_3-33)^2$, $f_4^\emptyset(T_4) = 4\cdot(T_4-45)^2$.

La suite des étapes effectuées sont les suivantes:

Etape 0: $C := \emptyset$, $\ell := 2$, $I_C := \{(1,2),(2,3),(3,4)\}$,
$\bar{T}_1 = 11$, $\bar{T}_2 = 24$, $\bar{T}_3 = 33$ et $\bar{T}_4 = 45$.

Etape 1: Vérification de l'arc $(1,2)$.
Puisque $\bar{T}_1 + t_{1,2} \leq \bar{T}_2$ $(11+10 \leq 24)$ alors aller à l'étape 2.

Etape 2: $\ell := 3$, aller à l'étape 1.

Etape 1: Vérification de l'arc $(2,3)$.
Puisque $\bar{T}_2 + t_{2,3} > \bar{T}_3$ $(24+10 > 33)$ alors aller à l'étape 3.

Etape 3: $f_3^C(T_3) = 3\cdot(T_3-33)^2 + 4\cdot((T_3-t_{23})-24)^2$,
$b_3^C = 25+10$,
$I_{C\cup\{2\}} = \{(1,3),(3,4)\}$,
$t_{1,3} = 10+10 = 20$.

Etape 4: Utilisant la formule donnée, dans la section suivante, pour les fonctions quadratiques, on obtient la valeur optimale suivante:
$\bar{T}_3 = (3\cdot 33 + 4\cdot(24+t_{23}))/(4+3) = 33.6$,
$C := \{2\}$,
aller à l'étape 1.

Etape 1: Vérification de l'arc $(1,3)$.
Puisque $\bar{T}_1 + t_{1,3} \leq \bar{T}_3$ $(11+20 \leq 33.6)$ alors aller à l'étape 2.

Etape 2: $\ell := 4$, aller à l'étape 1.

Etape 1: Vérification de l'arc $(3,4)$.
Puisque $\bar{T}_3 + t_{3,4} \leq \bar{T}_4$ $(33.6 + 10 \leq 45)$ alors aller à l'étape 2.

Etape 2: $\ell := 5$, aller à l'étape 5.

Etape 5: Puisque $C = \{2\}$, nous recalculons seulement $\bar{T}_2$.
$T_4^* = 45$,
$T_3^* = 33.6$,
$T_2^* = 33.6 - 10 = 23.6$,
$T_1^* = 11$.

Théorème 2: La complexité de l'algorithme est de l'ordre de n minimisations unidimentionnelles.

Preuve: L'étape 0 nécessite n minimisations unidimentionnelles, puisque le problème $(\bar{P})$ se décompose naturellement en n sous-problèmes indépendants. Puisque ℓ est compris entre 2 et n et qu'il ne décroît jamais, alors les étapes 2 n'impliquent que de l'ordre de n opérations élémentaires. A chaque exécution des étapes 3 et 4 la cardinalité de l'ensemble C augmente d'une unité. Puisque la cardinalité $|C|$ est comprise entre 1 et $n-1$ et qu'elle ne diminue jamais, alors chacune des étapes 3 et 4 est exécutée au plus $n-1$ fois. L'étape 3 n'implique que des opérations élémentaires et l'étape 4 implique une seule minimisation unidimentionnelle. A chaque itération, soit ℓ augmente ou soit $|C|$ augmente donc l'étape 1 est exécutée au plus $2 \cdot (n-1)$ fois et n'implique que des opérations élémentaires. Finalement, l'étape 5 implique au plus n opérations élémentaires. □

4.7. Spécialisation au cas de fonctions quadratiques

Les fonctions quadratiques sont un cas particulier des fonctions convexes. Pour ces fonctions quadratiques, le problème de minimisation unidimentionnelle se résout analytiquement. En effet, la fonction suivante:

$$f_i(T_i) = \alpha_i T_i^2 + \beta_i T_i + \gamma_i \,,\; a_i \leq T_i \leq b_i \tag{18}$$

atteint son minimum pour la valeur $\bar{T}_i$ suivante:

$$\bar{T}_i = \begin{cases} a_i & \text{si} \quad Z < a_i \\ Z & \text{si} \quad a_i \leq Z \leq b_i \\ b_i & \text{si} \quad Z > b_i \end{cases} \qquad \text{où } Z = \frac{-\beta_i}{2\alpha_i}. \tag{19}$$

Pour une fonction plus complexe qui consiste en une somme de fonctions quadratiques, telle:

$$f_i^C(T_i) = \sum_{k=\ell_i}^{i} f_k(T_i - t_{k,i}) \,, \quad a_i \leq T_i \leq b_i^C, \tag{20}$$

le minimum est atteint pour la valeur $\bar{T}_i$ suivante:

$$\bar{T}_i = \begin{cases} a_i & \text{si} \quad Z < a_i \\ Z & \text{si} \quad a_i \leq Z \leq b_i^C \\ b_i^C & \text{si} \quad Z > b_i^C \end{cases} \quad \text{où} \quad Z = \frac{\left(-\beta_i + \sum_{k=\ell_i}^{i-1} (2\alpha_k \cdot t_{k,i} - \beta_k)\right)}{2 \sum_{k=\ell_i}^{i} \alpha_k}. \tag{21}$$

Le calcul de Z peut être effectué plus efficacement en conservant des résultats intermédiaires au cours des itérations et ainsi la complexité du calcul de minimisation à l'étape 4 de l'algorithme est réduite. La complexité de l'algorithme devient de l'ordre de n opérations élémentaires.

4.8. Spécialisation au cas de fonctions linéaires

Lorsque les fonctions sont linéaires, la minimisation unidimentionnelle devient encore plus facile. En effet, la fonction suivante, qui est plus générale que celle de Sexton et Bodin (1985a):

$$f_i(T_i) = \beta_i T_i + \gamma_i \,,\; a_i \leq T_i \leq b_i \tag{22}$$

atteint son minimum pour la valeur $\bar{T}_i$ suivante:

$$\bar{T}_i = \begin{cases} a_i & \text{si} \quad \beta_i \geq 0 \\ b_i & \text{sinon} \end{cases} \tag{23}$$

Pour une somme de plusieurs fonctions de même type, telle:

$$f_i^C(T_i) = \sum_{k=\ell_i}^{i} f_k(T_i - t_{k,i}) \,, \quad a_i \leq T_i \leq b_i^C, \tag{24}$$

le minimum est atteint pour la valeur $\bar{T}_i$ suivante:

$$\bar{T}_i = \begin{cases} a_i & \text{si} \quad Z \geq 0 \\ b_i^C & \text{sinon} \end{cases} \qquad \text{où } Z = \sum_{k=\ell_i}^{i} \beta_k. \tag{25}$$

Les fonctions linéaires offrent un autre aspect intéressant puisque dans cette situation le problème dual du problème (P) se modélise comme un problème de flot à coût minimum. Réécrivons le problème (P) avec des fonctions linéaires.

$$(P) \qquad \min \;\; \sum_{i=1}^{n} (\beta_i T_i + \gamma_i) \tag{26}$$

sujet à

$$T_i + t_{i,i+1} \leq T_{i+1} \qquad i = 1, 2, \ldots, n-1 \; [-Y_{i,i+1}] \tag{27}$$

$$T_i \leq b_i \qquad i = 1, 2, \ldots, n \qquad [-Y_{id}] \tag{28}$$

$$-T_i \leq -a_i \qquad i = 1, 2, \ldots, n \qquad [-Y_{di}] \tag{29}$$

Les variables entre crochets indiquent les variables duales correspondantes. Ignorant les constantes γ_i, le dual se formule comme un problème de flot à coût minimum comme suit:

$$(D) \qquad -\min \quad -\sum_{i=1}^{n-1} Y_{i,i+1} \cdot t_{i,i+1} + \sum_{i=1}^{n} Y_{id} \cdot b_i - \sum_{i=1}^{n} Y_{di} \cdot a_i \qquad (30)$$

sujet à

$$Y_{i,i+1} + Y_{id} - Y_{i-1,i} - Y_{di} = -\beta_i \quad i = 1, 2, \ldots, n \quad [T_i] \qquad (31)$$

$$Y_{01} = 0, \quad Y_{n,n+1} = 0 \qquad (32)$$

$$Y_{i,i+1} \geq 0 \qquad i = 1, 2, \ldots, n-1 \qquad (33)$$

$$Y_{id} \geq 0\,,\ Y_{di} \geq 0 \qquad i = 1, 2, \ldots, n \qquad (34)$$

Les variables entre crochets indiquent les variables primales correspondantes et la variable $Y_{i,j}$ représente la variable de flot du noeud i au noeud j.

Les conditions d'optimalité que l'on obtient sont les suivantes:

$Y_{id} > 0$ implique $T_i = b_i$ $\quad i = 1, \ldots, n;$

$Y_{di} > 0$ implique $T_i = a_i$ $\quad i = 1, \ldots, n;$

$Y_{i,i+1} > 0$ implique $T_i + t_{i,i+1} = T_{i+1}$ $\quad i = 1, 2, \ldots, n-1.$

Le lemme 3 prouve que $Y_{id} \cdot Y_{di} = 0, \quad i = 1, \ldots, n$, i.e. qu'au moins une des deux variables est nulle. Un résultat plus fort est énoncé au Théorème 3 où la variable qui est nulle est de plus spécifiée.

Lemme 3: Soit le problème de flot (D). Il existe une solution optimale telle que $Y_{id} \cdot Y_{di} = 0 \quad i = 1, \ldots, n.$

Preuve: $Y_{di} > 0$ implique $T_i = a_i$ et $Y_{id} > 0$ implique $T_i = b_i$. Si $a_i < b_i$ alors on a nécessairement $Y_{id} \cdot Y_{di} = 0$, la preuve se faisant par contradiction. Si $a_i = b_i$ alors on peut modifier la solution, pour obtenir $Y_{id} \cdot Y_{di} = 0$, en diminuant Y_{id} et Y_{di} de la quantité ϵ où $\epsilon = \min(Y_{id}, Y_{di})$. La nouvelle solution demeure réalisable et optimale. □

Théorème 3: Soit le problème de flot (D). Soit $L = \{i \mid \beta_i \leq 0\}$ et $G = \{i \mid \beta_i > 0\}$ alors il existe une solution optimale du problème (D) où $Y_{di} = 0 \quad \forall i \in L$ et $Y_{id} = 0 \quad \forall i \in G$.

Preuve : Soit une solution optimale du problème (D) tel que $\alpha = \sum_{i \in L} Y_{di} + \sum_{i \in G} Y_{id} + \sum_{i=1}^{n-1} Y_{i,i+1}$ est minimum. Si $\sum_{i \in L} Y_{di} + \sum_{i \in G} Y_{id} = 0$ alors le théorème est montré. Sinon, soient $L' = \{i \mid Y_{id} > 0, \beta_i \leq 0\}$ et $G' = \{i \mid Y_{di} > 0, \beta_i > 0\}$ et montrons que α n'est pas minimum. Supposons $L' \neq \emptyset$ et $k \in L'$ (On peut vérifier facilement que $n \notin L'$). Puisque $k \in L'$ alors $Y_{dk} > 0$, $\beta_k \leq 0$, $Y_{kd} = 0$ (par le lemme 3) et $Y_{k,k+1} \geq Y_{dk} > 0$ ($k^{ième}$ contrainte: $Y_{k,k+1} - Y_{k-1,k} - Y_{dk} = \beta_k > 0$). Soit $\epsilon = Y_{dk}$. La nouvelle solution obtenue en diminuant Y_{dk} et $Y_{k,k+1}$ de ϵ et en augmentant $Y_{d,k+1}$ de ϵ conserve la faisabilité. Puisque $a_k + t_{k,k+1} \leq a_{k+1}$, l'objectif est modifié de la quantité $\epsilon \cdot t_{k,k+1} + \epsilon \cdot a_k - \epsilon \cdot a_{k+1} \leq 0$, donc la solution est au moins aussi bonne. La variation sur α est de $\epsilon - \epsilon - \epsilon < 0$ ce qui implique une contradiction. Donc $L' = \emptyset$. Par un raisonnement semblable on peut montrer que $G' = \emptyset$ en utilisant le fait que $b_k + t_{k,k+1} \leq b_{k+1}$. □

Le théorème 3 permet de simplifier le réseau du problème de flot. On obtient un réseau de $n+1$ noeuds; un noeud pour chaque variable T_i, $i = 1, 2, \ldots, n$, avec une demande β_i et un noeud artificiel d avec une disponibilité $\beta_d = \sum_{i=1}^{n} \beta_i$. Le réseau contient $2n-1$ arcs; ces arcs sont définis dans la Tableau 1. Notons que le réseau obtenu est différent de celui de Sexton et Bodin (1985a).

Tableau 1: Description des arcs du réseau.

flot de	flot à	condition d'existence	nom de la variable	coût par unité de flot
i	d	si $\beta_i \leq 0$	Y_{id}	b_i
d	i	si $\beta_i > 0$	Y_{di}	$-a_i$
i	$i+1$		$Y_{i,i+1}$	$-t_{i,i+1}$

La figure 2 présente un exemple de la structure du réseau. Pour cet exemple nous avons choisi $N = \{1, 2, 3, 4, 5\}$, $\beta_1 < 0$, $\beta_2 > 0$, $\beta_3 < 0$, $\beta_4 > 0$, $\beta_5 > 0$ et $\beta_1 + \beta_2 + \beta_3 + \beta_4 + \beta_5 = \beta_d > 0$. A chacun des arcs est associé le coût de l'arc. A chaque noeud la quantité β de flot produit ou consommé est indiquée.

Les variables duales associées aux noeuds du réseau sont les variables T_i du problème original. Pour le noeud supplémentaire d on définit la variable T_d comme étant égale à zéro. Il est intéressant de constater que notre algorithme effectue les mêmes itérations que l'algorithme du simplexe, en débutant avec la base composée des variables Y_{id} si $\beta_i \leq 0$ et Y_{di} si $\beta_i > 0$, $\forall i \in N$ et en utilisant comme critère d'entrée la variable $Y_{i,i+1}$ de plus petit indice i dont le coût relatif est négatif. Il est facile de vérifier que les coûts relatifs des arcs associés aux variables Y_{id} et Y_{di} seront toujours positifs ou nuls. En fait, les seuls arcs qui peuvent entrer dans la base sont les arcs $Y_{i,i+1}$ et l'entrée d'un de ces arcs dans la base correspond à fixer à égalité la contrainte $(i, i+1)$ du problème (P). Les deux approches correspondent à décider séquentiellement quelle contrainte est fixée à égalité et les décisions prises ne sont pas remises en cause.

4.9. Extensions

On peut généraliser le modèle en ajoutant un coût linéaire sur le temps d'attente. Cette généralisation est intéressante pour les situations où l'attente présente un inconvénient. Le nouvel objectif devient:

$$\min \sum_{i=1}^{n} f_i(T_i) + g_1(T_1 - a_1) + \sum_{i=2}^{n} g_i\big(T_i - (T_{i-1} + t_{i-1,i})\big) \qquad (35)$$

où $g_i(x)$ est une fonction de coût du temps d'attente x au noeud i et s'écrit comme:

$$g_i(x) = \begin{cases} \Gamma_i \cdot x & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases} \qquad (36)$$

Puisqu'à l'optimum les contraintes (2) et (3) sont respectées, i.e., $T_i - (T_{i-1} + t_{i-1,i}) \geq 0$ et $T_i - a_i \geq 0$, alors on peut remplacer $g_i(x)$ par $\Gamma_i \cdot x$. L'objectif devient:

$$\min \sum_{i=1}^{n} f_i(T_i) + \Gamma_1 \cdot (T_1 - a_1) + \sum_{i=2}^{n} \Gamma_i \cdot \left(T_i - (T_{i-1} + t_{i-1,i})\right) \tag{37}$$

et après regroupement on obtient:

$$\min \sum_{i=1}^{n-1} [f_i(T_i) + (-\Gamma_{i+1} + \Gamma_i) \cdot T_i] + f_n(T_n) + \Gamma_n \cdot T_n - \sum_{i=2}^{n} \Gamma_i \cdot t_{i-1,i} \tag{38}$$

Le dernier terme est une constante et peut être éliminé sans changer la solution optimale. Finalement réécrivons l'objectif en termes des fonctions f_i' définies comme suit:

$$f_i'(x) = \begin{cases} f_i(x) + (-\Gamma_{i+1} + \Gamma_i) \cdot x & \text{si } i = 1, \ldots, n-1 \\ f_n(x) + \Gamma_n \cdot x & \text{si } i = n \ . \end{cases} \tag{39}$$

L'objectif devient:

$$\min \quad \sum_{i=1}^{n} f_i'(T_i). \tag{40}$$

Puisque $f_i(x)$ et $\Gamma_i \cdot x$ sont convexes alors $f_i'(x)$ l'est aussi. L'objectif obtenu est de la même forme que celui présenté à la section 2 et l'algorithme de la section 6 s'applique.

Jusqu'à présent, nous avons considéré les variables T_i aux noeuds comme étant des variables continues sur l'intervalle $[a_i, b_i]$. Une deuxième généralisation consiste à considérer T_i comme une variable discrète. Le problème demeure facile à résoudre; la minimisation unidimentionnelle se fait toujours de la même façon sauf que la valeur $\bar{T}_\ell$ obtenue à l'étape 4 est modifiée comme suit:

$$\bar{T}_\ell := \begin{cases} \lceil \bar{T}_\ell \rceil & \text{si } f_\ell^{C \cup \{k\}}(\lceil \bar{T}_\ell \rceil) < f_\ell^{C \cup \{k\}}(\lfloor \bar{T}_\ell \rfloor), \\ \lfloor \bar{T}_\ell \rfloor & \text{sinon,} \end{cases} \tag{41}$$

c'est-à-dire qu'on choisit la meilleure valeur discrète qui est soit l'entier précédant ou soit l'entier suivant de $\bar{T}_\ell$. Notons que l'on peut également adapter l'algorithme pour des situations plus générales où T_i prend ses valeurs dans un ensemble W_i fermé et borné.

4.10. Conclusion

Nous avons présenté un algorithme pour le problème de l'optimisation de l'horaire d'un véhicule visitant une suite de noeuds dans un ordre prédéterminé. Ce problème se retrouve comme dernière étape ou comme sous-problème (décomposition de Benders) dans la plupart des approches de résolution, pour les problèmes d'optimisation d'itinéraires et d'horaires de véhicules. Nous avons montré que la complexité de l'algorithme, en nombre de minimisations unidimentionnelles, est de l'ordre du nombre de noeuds pour des fonctions convexes et la complexité se compte en nombre d'opérations élémentaires pour des fonctions quadratiques et linéaires.

La généralisation aux fonctions d'inconvénients convexes, avec des intervalles de temps, offre un aspect plus intéressant que le cas linéaire puisqu'elle permet de générer des solutions non-extrémistes. Une extension a été proposée pour permettre d'inclure des coûts linéaires sur le temps d'attente et une autre pour le cas où les variables de temps sont discrètes. Ces extensions n'impliquent pas de modifications à l'algorithme proposé.

4.11 Références

BODIN L., B. GOLDEN, A. ASSAD and M. BALL (1983) Routing and Scheduling of Vehicles and Crews: The State of the Art, *Comput. Opns. Res.* 10, 62–212.

DESROCHERS M., J.K. LENSTRA, M.W.P. SAVELSBERGH and F. SOUMIS (1988) Vehicle Routing with Time Windows: Optimization and Approximation, *TIMS Studies in Management Science* 16, 65–84.

DESROSIERS J., F. SOUMIS and M. DESROCHERS (1984) Routing with Time Windows by Colum Generation, *Networks*, 14, 545–565.

DESROSIERS J., Y. DUMAS and F. SOUMIS (1986) A Dynamic Programming Solution of the Large-Scale Single-Vehicle Dial-a-Ride Problem with Time Windows, *The American Journal of Mathematical and Management Sciences* 6, 301–325.

DESROSIERS J., Y. DUMAS and F. SOUMIS (1988) The Multiple Vehicle DIAL-A-RIDE Problem, *Lecture Notes in Economics and Mathematical System 308: Computer-Aided Transit Scheduling*, J.R. Daduna, A. Wren (Editors), 1, Springer-Verlag, 5–27.

DUMAS, Y. (1985) Confection d'itinéraires de véhicules en vue du transport de plusieurs origines à plusieurs destinations, *Thèse de maîtrise*, Publication # 434, Centre de Recherche sur les Transports, Université de Montréal,Canada, H3C 3J7.

KOLEN A., A. RINNOOY KAN and H. TRIENEKENS (1987) Vehicle Routing with Time Windows, *Operations Research* 35, 266–273.

PSARAFTIS H. (1983) A Dynamic Programming Solution to the Single-Vehicle Many-to-Many Immediate Request Dial-A-Ride Problem, *Transportation Science*, 17, 351–357.

SEXTON T. and L. BODIN (1985a) Optimizing Single Vehicle Many-to-Many

Operations with Desired Delivery Times: I. Scheduling, *Transportation Science* 19, 378–410.

SEXTON T. and L. BODIN (1985b) Optimizing Single Vehicle Many-to-Many Operations with Desired Delivery Times: II. Routing, *Transportation Science* 19, 411–435.

SEXTON T. and Y. CHOI (1986) Pick-up and Delivery of Partial Loads with Time Windows, *The American Journal of Mathematical and Management Sciences* 6, 369–398.

SOLOMON M. and J. DESROSIERS (1988) Time Window Constrained Routing and Scheduling Problems, *Transportation Sciences* 22, 1–13.

SOUMIS F., J. DESROSIERS and M. DESROCHERS (1985) Optimal Urban-Bus Routing with Scheduling Flexibilities, in *Lecture Notes in Control and Information Sciences* 59, Springer-Verlag, 155–165.

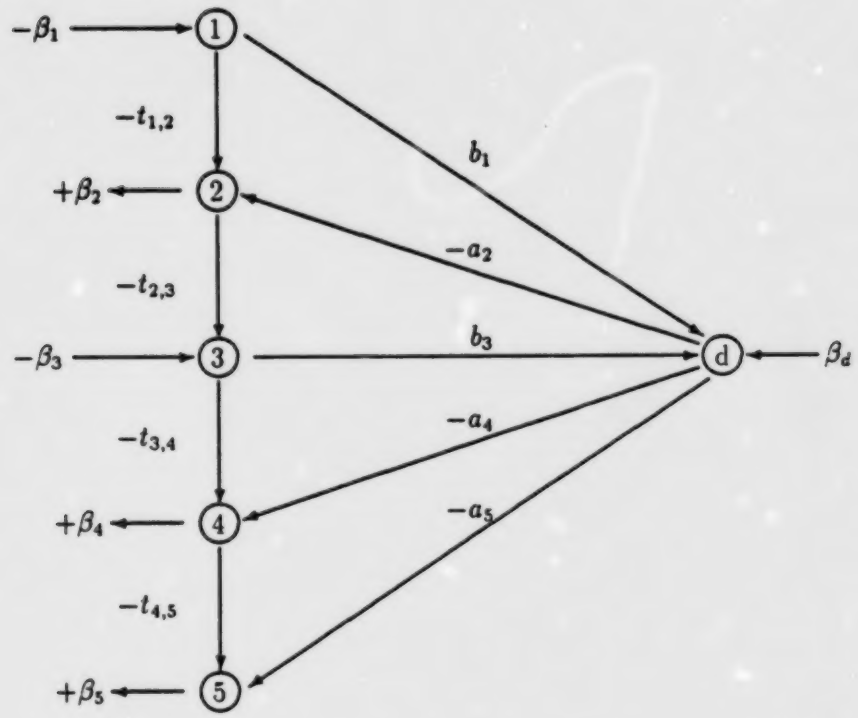


Figure 2: Structure du problème de flot à coût minimum.

5

Expérimentation du prototype DARSY

Nous avons développé un prototype appelé DARSY (Dial-A-Ride SYstem) basé sur l'approche exposée dans cette thèse. Nous évaluons ses performances sur des données provenant de la journée d'opération du lundi le 25 novembre 1985 à la Wheel-Trans Service de Toronto (WTST). Le problème comprend 1890 requêtes formulées au moins un jour avant la journée d'opération ainsi qu'un ensemble de 133 pièces de travail.

Ce chapitre comporte deux sections principales. La première présente le prototype DARSY et la deuxième l'expérimentation.

5.1 Le prototype DARSY

Le prototype DARSY a été développé pour permettre de résoudre des problèmes de confection d'itinéraires et d'horaires de grande taille (par exemple, plus de 2000 requêtes et plus de 150 véhicules) dans le contexte du transport adapté de façon à fournir des solutions de qualité supérieure à celles fournies par les approches déjà existantes.

Cette section décrit la procédure générale d'optimisation de DARSY ainsi que le découpage géographique et temporel, utilisé à une étape de la procédure générale, pour résoudre un m-TSPTW de grande taille.

Le tableau 1 illustre les principales étapes de la procédure générale. A chacune des étapes, on mentionne la méthodologie utilisée et les conditions pour son bon fonctionnement. Les étapes 1 à 4 correspondent à notre phase de répartition des requêtes entre les véhicules ("clustering"), l'étape 5 correspond à la phase d'ordonnancement des arrêts de chaque véhicule ("routing") et l'étape 6 est associée à la phase de confection de l'horaire de chaque véhicule ("scheduling").

Tableau 1: Procédure générale de DARSY.

Étapes	Méthodologies	Conditions
1 – Construction d'une solution initiale	– Insertion en parallèle	
2 – Optimisation des itinéraires un à un (1-PDPTW)	– Programmation dynamique	– Solution initiale ("clustering") – Temps euclidiens – Problème très contraint
3 – Fabrication des tâches	– Morcellement des itinéraires	– Solution initiale
4 – Optimisation des itinéraires pour répartir les tâches entre les véhicules (m-TSPTW de grande taille)	– Découpage temporel et géographique – Génération de colonnes – "Branch and bound"	– Problème contraint
5 – Optimisation des itinéraires un à un (1-PDPTW)	– Programmation dynamique	– Solution initiale ("clustering") – Temps euclidiens – Problème très contraint
6 – Optimisation de l'horaire des itinéraires un à un	– Méthode duale de complexité linéaire	

La première étape construit une solution initiale à l'aide de l'algorithme modifié de Roy et al. (1985). L'algorithme est une procédure d'insertion en parrallèle procédant en deux phases. La première tente d'insérer les demandes par groupe de demandes voisines et la seconde tente d'insérer une à une les demandes non insérées de la première phase. L'algorithme initialise une nouvelle pièce de travail uniquement lorsque nécessaire et tente de minimiser le déplacement en insérant selon des critères de détour minimum. Principalement, l'algorithme de Roy et al. (1985) a été modifié pour les raisons suivantes: 1) pour accepter des matrices de distances plus volumineuses (au lieu de conserver la matrice en mémoire, la matrice est conservée dans un fichier en accès direct), 2) pour utiliser une fonction de temps tenant compte de l'heure du déplacement et du temps de service, 3) pour tenir compte des configurations multiples de la capacité des véhicules, ainsi que 4) pour choisir différemment les pièces de travail . Notons que l'algorithme ne permet aucune violation des contraintes de capacité et de fenêtres horaires.

La deuxième étape réoptimise un à un les itinéraires de la solution initiale à l'aide d'un algorithme exact de programmation dynamique de façon à minimiser le déplacement. Cet algorithme de programmation dynamique vers l'avant est conçu pour résoudre le problème de cueillette et livraison d'un seul véhicule, soit le 1-PDPTW où les contraintes de fenêtres horaires et/ou de capacité sont restrictives. La description de cet algorithme se retrouve au chapitre 3.

La troisième étape regroupe les requêtes en tâches. Les requêtes sont regroupées en tâches en morcelant les itinéraires de l'étape précédente aux endroits où le véhicule se déplace sans passager. Une tâche est définie par une portion d'itinéraire et se caractérise par une fenêtre horaire (pour débuter la portion d'itinéraire), une durée maximum et des charges maximum atteintes. Notons que les charges sont multidimensionnelles et qu'il n'existe pas d'ordre total. Il peut ainsi y avoir plusieurs charges maximum atteintes, par exemple 3 ambulants–2 fauteuils ou 2 ambulants–3 fauteuils.

La quatrième étape résout un m-TSPTW pour répartir les tâches entre les véhicules. Lorsque la taille du problème est petite (moins de 150 tâches) le problème est résolu directement à l'aide d'un algorithme de génération de colonnes

et d'une procédure de "branch-and-bound" qui sont décrit et testés au chapitre 2. Lorsque la taille du problème est considérable on utilise un découpage temporel et géographique. Ce découpage consiste tout simplement de remplacer l'optimisation du problème m-TSPTW global par l'optimisation d'une suite de problèmes m-TSPTW plus petits et plus faciles à résoudre. Ce découpage est décrit à la section 5.1.1 suivante.

La cinquième étape réoptimise un à un les itinéraires de la solution produite à l'étape 4. L'algorithme utilisé est celui de l'étape 2.

La sixième étape complète la procédure générale par l'utilisation d'un algorithme exact de détermination des heures de services aux arrêts des demandes. L'algorithme minimise par une méthode duale de complexité linéaire des fonctions d'inconvénients convexes définies sur les fenêtres horaires des arrêts. Cette méthode est présentée au chapitre 5.

Trois types de conditions sont mentionnés au tableau 1:

a) Une solution initiale doit être disponible. Cette condition est satisfaite par l'étape 1.

b) Les temps de déplacement doivent être euclidiens. Lorsque les distances sont euclidiennes, toute fonction croissante de transformation des distances en temps fournit des temps de déplacement euclidiens. Par contre, lorsque la fonction de transformation comporte des arrondissements numériques, les temps ne sont plus euclidiens.

c) Les problèmes doivent être contraints, i.e. les contraintes doivent être suffisamment fortes pour réduire significativement l'exploration combinatoire. Cette condition est satisfaite dès que les fenêtres horaires sont assez étroites.

5.1.1 Découpage temporel et géographique

La quatrième étape de la procédure générale résout un m-TSPTW pour répartir les tâches entre les véhicules. *Lorsque la taille du problème est petite* (moins de 150 tâches) le problème est résolu directement à l'aide d'un algorithme de génération de colonnes et d'une procédure de "branch-and-bound" tels que décris et testés au chapitre 2. Lorsque la taille du problème est considérable on utilise un découpage temporel et géographique. Ce découpage consiste tout simplement à remplacer l'optimisation du problème m-TSPTW global par l'optimisation d'une suite de problèmes m-TSPTW plus petits et plus faciles à résoudre. Chacun d'eux est défini pour une période et une région données. A partir d'une solution heuristique initiale, l'optimisation du problème m-TSPTW global correspond à remettre en question tous les inter-tâches de cette solution. En fixant des portions d'itinéraires à l'extérieur de la période et la région considérées, on réduit la combinatoire et on concentre l'optimisation sur la période et la région choisies.

Pour des problème moyennement gros (moins de 500 tâches) le découpage temporel est suffisant. Il consiste à découper l'horizon du temps en $k-1$ intervalles et à faire correspondre à chaque intervalle un problème d'optimisation dont les portions d'itinéraires avant et après l'intervalle sont fixées et remplacées par de nouvelles tâches. Chacune des nouvelles tâches regroupe les tâches d'une portion d'itinéraire fixée. La figure 1 présente pour une période donnée l'ensemble des tâches d'un problème à résoudre. La figure illustre trois itinéraires où les 25 tâches originales sont remplacées par 14 nouvelles tâches. Notons que ce découpage permet de conserver une vue globale du problème.

Puisqu'une tâche i a une fenêtre de temps $[a_i, b_i]$ pour le début de son exécution et une durée d_i, il n'est pas trivial de choisir l'ensemble des tâches à considérer dans la période considérée. Soit $\beta_1, \beta_2, \ldots, \beta_k$ les points de coupure de l'horizon et soit l'intervalle de temps $[\beta_{i-1}-\epsilon, \beta_i+\epsilon]$ associé au $(i-1)^{ème}$ problème d'optimisation. Le paramètre ϵ sert au recouvrement des intervalles de temps pour assurer une meilleure optimisation de la suite des problèmes.

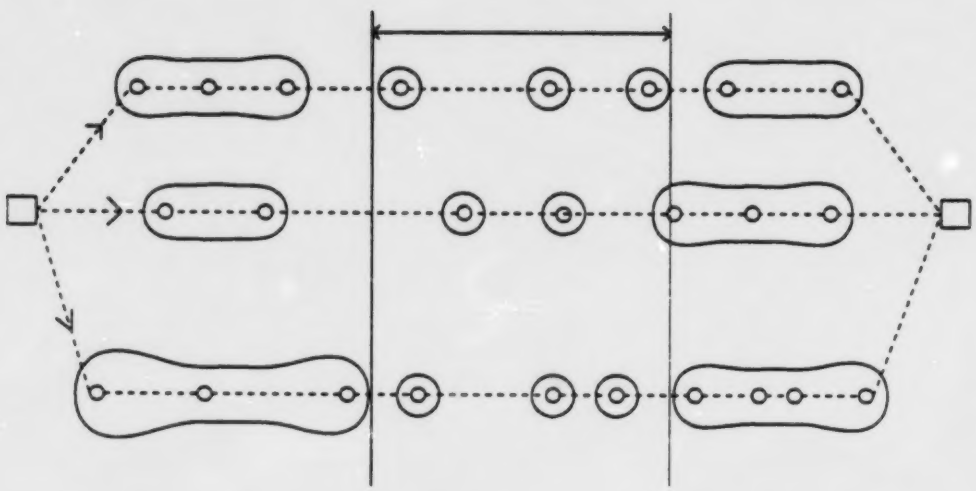


Figure 1: Découpage temporel d'une période.

Voici 5 critères qui pouraient être utilisés pour l'acceptation de la tâche i dans l'intervalle $[\beta_{i-1} - \epsilon, \beta_i + \epsilon]$:

1- la tâche i peut débuter dans l'intervalle, i.e.
$[a_i, b_i] \cap [\beta_{i-1} - \epsilon, \beta_i + \epsilon] \neq \phi$,

2- la tâche i peut terminer dans l'intervalle, i.e.
$[a_i + d_i, b_i + d_i] \cap [\beta_{i-1} - \epsilon, \beta_i + \epsilon] \neq \phi$,

3- la tâche i peut débuter et terminer dans l'intervalle,

4- la tâche i peut débuter ou terminer dans l'intervalle,

5- au moins $\gamma \cdot (100)$ % de la durée de la tâche peut être effectuée dans l'intervalle, i.e. $\min(\beta_i + \epsilon, b_i + d_i) - \max(a_i, \beta_{i-1} - \epsilon) \geq \gamma \cdot d_i$.

Nous avons choisi le $5^{ème}$ critère pour les raisons suivantes: a) le critère est plus général, b) l'ensemble des tâches acceptées assure une meilleure continuité entre les périodes adjacentes et c) l'ensemble des tâches acceptées représente bien l'activité des véhicules dans la période de temps considérée. Notons que pour γ petit le $5^{ème}$ critère s'apparente au $4^{ème}$ critère et pour γ grand le $5^{ème}$ critère s'apparente au $3^{ème}$ critère. Nous avons utilisé ce critère avec $\gamma = 0.75$

Une procédure automatique détermine les points de coupure $\beta_1, \ldots, \beta_k$ de façon à obtenir des tailles de problème raisonnables et à peu près égales (environ 100 tâches) dans chacune des périodes.

Lorsque le problème m-TSPTW global est très gros (plus de 500 tâches), le découpage temporel n'est plus suffisant et il est nécessaire de faire appel au découpage géographique. Le découpage géographique est justifié lorsqu'on suppose que l'activité des véhicules d'une région influence peu l'activité des véhicules d'une région plus éloignée. La procédure de découpage géographique procède comme suit: pour une période de temps donnée, la ville est découpée d'Ouest en Est puis d'Est en Ouest comme présenté à la figure 2. Pour chaque région d'un intervalle de temps donné, on fait correspondre un problème d'optimisation dont les portions d'itinéraires à l'extérieur de la région et de l'intervalle de temps sont fixées et remplacées par de nouvelles tâches. Ainsi on conserve une vue globale du problème. Une procédure automatique supplémentaire détermine la taille et le recouvrement des régions en considérant la taille désirée des petits problèmes m-TSPTW. Il existe plusieurs façons de procéder au découpage des régions comme il existe plusieurs façons de déterminer les tâches appartenant à une région donnée. Nous avons choisi d'inclure une tâche dans une région géographique si $\gamma \cdot (100)$ % du déplacement direct du point de départ au point d'arrivé de la tâche se situe dans la région. Pour ce critère, nous avons utilisé $\gamma = 1.0$.

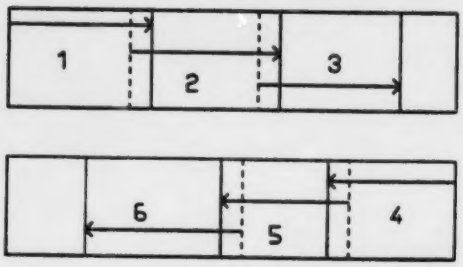


Figure 2: Découpage géographique d'Ouest en Est puis d'Est en Ouest d'un schéma d'une ville.

5.2 Expérimentation

Nous évaluons les performances de DARSY sur des données provenant de la journée d'opération du lundi le 25 novembre 1985 à la WTST. La section 5.2.1 présente l'ensemble des données, la section 5.2.2 décrit le plan d'expérience, la section 5.2.3 présente les solutions initiales et la dernière section présente 11 optimisations différentes.

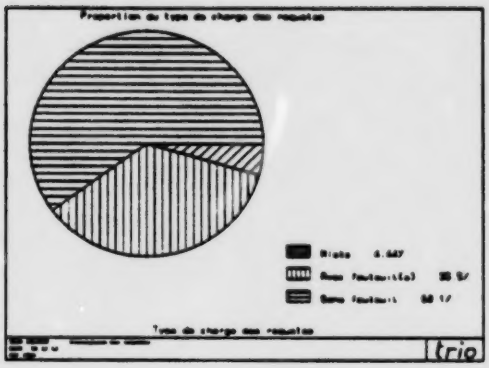
5.2.1 Les données

Le problème est constitué de 1890 requêtes (2107 passagers) et de 133 pièces de travail.

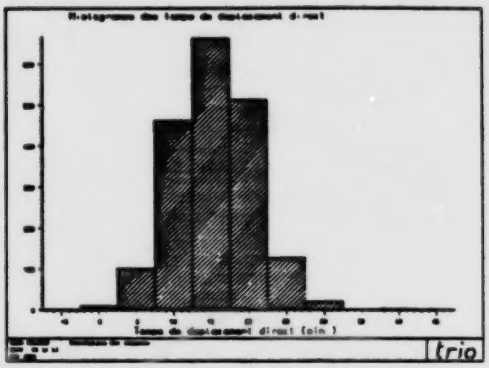
Les figures 3(a) à 3(d) illustrent les caractéristiques de ces requêtes. A la figure 3(a) le type de charge des requêtes est présenté. Les requêtes sont divisées, selon leur charge, en trois catégories. La première est constituée des requêtes sans fauteuil roulant (60 %), la seconde des requêtes avec seulement des fauteuil(s) roulant(s) (35.5 %) et la troisième des requêtes impliquant à la fois au moins une personne avec un fauteuil roulant et une personne sans fauteuil roulant (4.5 %).

A la figure 3(b) les temps de déplacement direct de chacune des requêtes y sont représentés. La majorité des temps de déplacement direct se situe entre 5 et 25 minutes et le temps de déplacement maximum est de 40 minutes. Les temps de 0 minute correspondent à des déplacements entre deux adresses voisines dans la même zone pour lesquels la matrice des distances corrigées fournit une distance de déplacement nulle. Aux figures 3(c) et 3(d), la largeur des fenêtres aux arrêts d'origine et de destination y est représentée. Pour les arrêts d'origine, la largeur des fenêtres s'étend de 60 à 130 minutes et pour les arrêts de destination, de 75 à 145 minutes. Cette grande flexibilité peut être acceptable dans le contexte où les heures de prise sont confirmées par téléphone et qu'une flexibilité de 10 à 15 minutes est alors accordée sur l'heure confirmée.

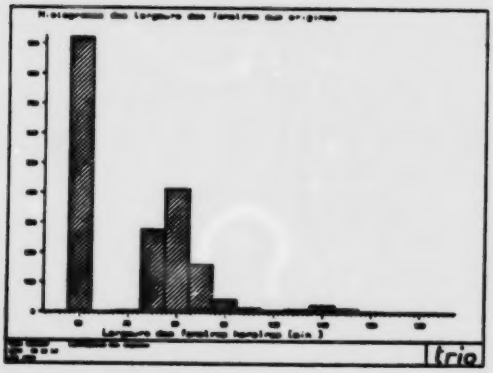
Les 133 pièces de travail proviennent des 116 pièces de travail utilisées par la WTST auxquelles nous avons ajouté 17 pièces qui sont des duplicatas de 17 d'entres elles. On distinque chaque pièce de travail par une lettre (sauf pour les duplicatas) suivi d'un numéro tels: b001, c117, s253, t350. Les duplicatas sont identifiés par la lettre z suivi de l'identification de la pièce de travail duplicatée, par exemple: zb01, zb05, zb60. Les pièces de travail dont l'identification inclut les lettres t ou s sont associées à un véhicule de capacité maximale de 4 positions assises et d'aucune position pour les fauteuils roulants. Celles identifiées par un b ou un c ont une capacité maximale de 6 positions assises, de 5 positions pour les fauteuils roulants et de 10 positions au total. Par exemple, pour ces dernières, on peut avoir 6 personnes en position assise et 4 personnes dans leur fauteuil ou bien 5-5 mais pas 6-5. La figure 4 illustre l'ensemble des pièces de travail. On distingue 16 pièces de travail différentes. L'amplitude maximale est de 13 heures, l'amplitude minimale de 3 heures et le total des heures disponibles pour l'ensemble des pièces de travail totalise 1130 heures.



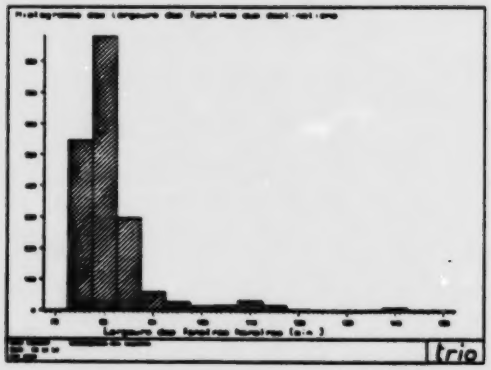
(a) Type de charge des requêtes



(b) Temps de déplacement direct (min.)



(c) Largeur des fenêtres horaires aux origines



(d) Largeur des fenêtres horaires aux destinations

Figure 3: Description des requêtes

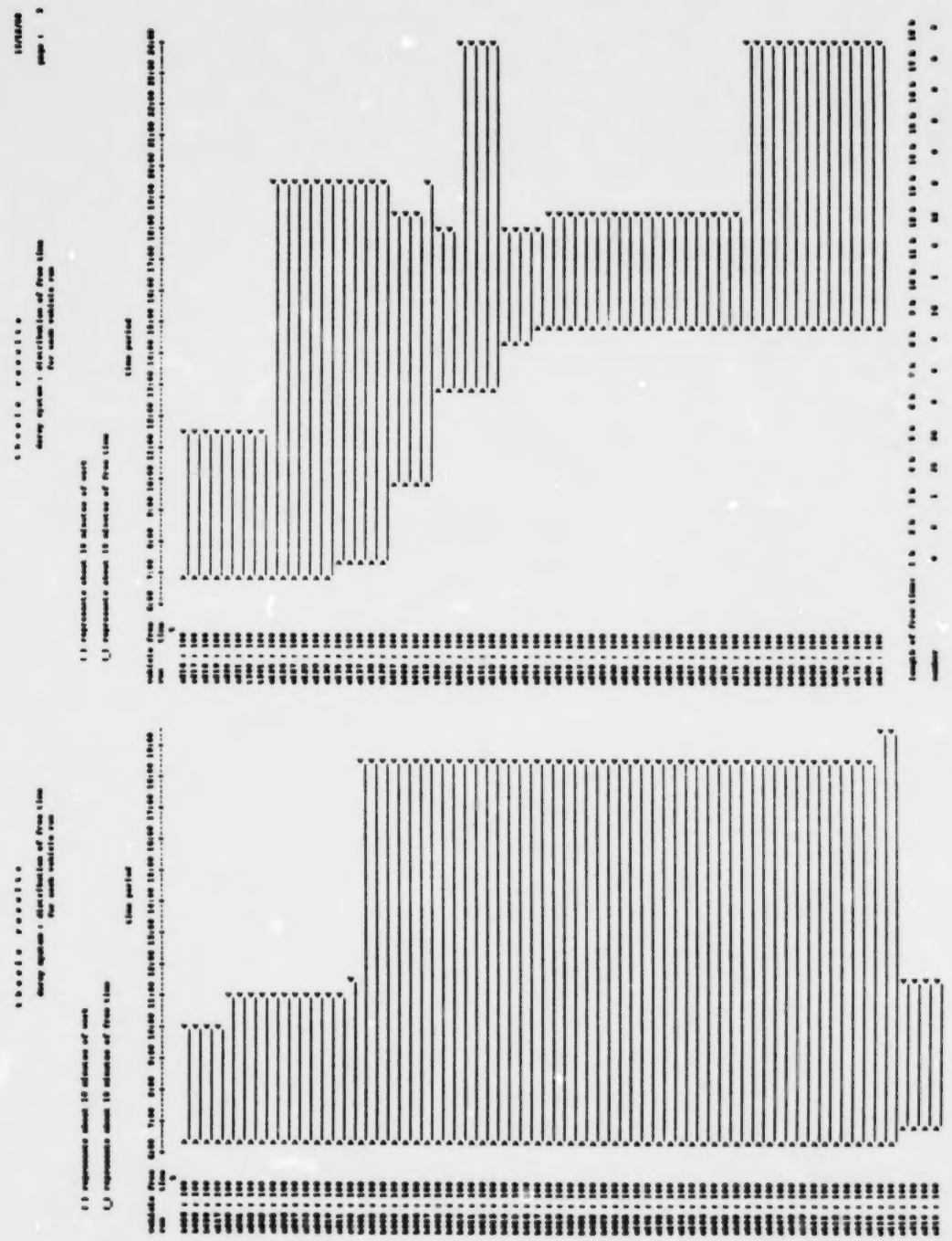


Figure 4: Ensemble des pièces de travail

5.2.2 Plan d'expérience

L'objectif était celui de développer un prototype pouvant résoudre des problèmes de grande taille de façon à fournir des solutions de qualité supérieure à celles fournies par les approches déjà existantes. Nous sommes en mesure de montrer que DARSY peut résoudre des problèmes de grande taille puisque les données utilisées comporte 1890 requêtes et 133 véhicules. Par contre, il est plus difficile de montrer qu'il fournit des solutions de qualité supérieure à celles des approches déjà existantes. Toutefois, on comparera les résultats du prototype DARSY avec les solutions produites par l'algorithme d'insertion en parallèle de Roy et al. (1985) et avec la solution manuelle produite par les répartiteurs de la WTST.

Notre plan d'expérience comporte 11 optimisations. Ces optimisations permettent d'évaluer la capacité d'optimisation de DARSY pour différentes situations, en faisant varier les points suivants:

- **La solution initiale.** La solution initiale est soit produite à l'étape 1 de la procédure générale, ou correspond à la solution de la WTST ou à la solution finale de DARSY (dans le cas de ré-optimisation).

- **La fonction de temps de déplacement.** La fonction de temps de déplacement fournit soit des temps aux 5 minutes ou des temps aux minutes. A la WTST les répartiteurs utilisent des temps évalués aux 5 minutes. Nous avons donc contruit une fonction évaluant les temps de déplacement aux 5 minutes et une autre, plus précise, évaluant les temps aux minutes.

- **La flexibilité accordée sur les fenêtres horaires des tâches.** Elle est soit limitée à 15 minutes ou à 30 minutes. Une flexibilité plus stricte permet une optimisation plus rapide.

- **La fonction économique du m-TSPTW.** La fonction économique est soit la minimisation du déplacement ou la maximisation des durées des périodes libres. La fonction économique de minimisation du nombre de

véhicules est difficile à considérer dans DARSY parce que l'optimisation se fait plutôt à partir de pièces de travail d'amblitudes variables. Par contre, nous verrons que la maximisation de la durée des périodes libres offre un compromis intéressant entre la minimisation du déplacement et la minimisation du nombre de pièces de travail.

Les solutions initiales sont identifiées par TOR ou ROY d'où l'on fait suivre -1m ou -5m suivant que la fonction de temps de déplacement utilisée génère des temps aux minutes ou aux 5 minutes respectivement. L'identification TOR indique que la solution a été produite manuellement par la WTST et l'identification ROY indique que la solution a été produite par l'algorithme modifié de ROY et al. (1985) (étape 1 de la procédure générale de DARSY).

Utilisant la notation du chapitre 1, la fonction économique utilisée pour le m-TSPTW est la suivante:

$$\max \sum_{v \in V} \sum_{i \in N} \sum_{j \in N} L_{ij}^{\alpha} \cdot X_{ij}^{v}$$

où $L_{ij} = T_j - T_i - t_{ij}$. Lorsque le véhicule v va directement de la tâche i à la tâche j, L_{ij} correspond à la durée de la période libre du véhicule entre ces deux tâches. Les valeurs L_{ij} sont évaluées dynamiquement dans l'algorithme de plus court chemin. Pour $\alpha = 1$ la fonction économique correspond à minimiser le temps de déplacement puisque pour chaque véhicule v :

$$\sum_{i \in N} \sum_{j \in N} (T_j - T_i - t_{ij}) \cdot X_{ij}^{v} = T_{2n+1}^{v} - T_0^{v} - \sum_{i \in N} \sum_{j \in N} t_{ij} \cdot X_{ij}^{v}$$

où $T_{2n+1}^{v} - T_0^{v}$ est une constante représentant l'amplitude de la pièce de travail associée au véhicule v. Lorsque α est grand la fonction économique accorde plus de poids aux grandes périodes libres, ce qui tend à libérer les grandes pièces de travail donc à minimiser leur nombre. Nous avons utilisé $\alpha = 1$ pour la minimisation du déplacement et $\alpha = 1.2$ pour la maximisation des durées des périodes libres. Cette dernière valeur correspond à notre désir de priviligier des périodes libres de grande durée sans pour autant négliger la minimisation du déplacement.

Le tableau 2 fournit notre plan d'expérience complet. Les 11 optimisations sont identifiées par des lettres majuscules. Les solutions finales C, F et I

correspondent aux ré-optimisation des solutions B, E et H respectivement. Les solutions du groupe utilisant une fonction de temps de déplacement aux 5 minutes ont été obtenues en utilisant les solutions initiales spécifiées et en effectuant les étapes 3, 4 et 6 de la procédure générale de DARSY. Les étapes 2 et 5 n'ont pas été effectuées puisque les temps de déplacement aux 5 minutes ne sont pas euclidiens. Pour les solutions du groupe utilisant une fonction de temps de déplacement aux minutes, elles ont été obtenues en utilisant les solutions initiales spécifiées et en effectuant les étapes 2 à 6 de la procédure générale de DARSY.

Tableau 2: Plan d'expérience

Fonction de temps de déplacement	**Flexibilité maximale des fenêtres horaires**	**Solution initiale**	**Objectif du *m*-TSPTW(*)**	**Solution finale**
aux 5 min.	15 min.	TOR-5m	$\alpha = 1$	A
			$\alpha = 1.2$	B
		B	$\alpha = 1.2$	C
		ROY-5m	$\alpha = 1$	D
			$\alpha = 1.2$	E
		E	$\alpha = 1.2$	F
aux min.	15 min.	ROY-1m	$\alpha = 1$	G
			$\alpha = 1.2$	H
		H	$\alpha = 1.2$	I
	30 min.	ROY-1m	$\alpha = 1$	J
			$\alpha = 1.2$	K

(*) $\alpha = 1.$ vaut pour la minimisation du temps de déplacement
$\alpha = 1.2$ vaut pour la maximisation des durées des périodes libres

5.2.3 Les solutions initiales

Cette section présente les trois principales solutions initiales utilisées dans nos tests ainsi que la description des tâches résultant du morcellement des itinéraires de ces solutions initiales.

Le tableau 3 fournit la description des trois principales solutions initiales. Les solutions initiales sont identifiées par TOR ou ROY suivi d'un de -1m ou -5m suivant que la fonction de temps de déplacement utilisée est aux minutes ou aux 5 minutes respectivement. L'identification TOR indique que la solution a été produite manuellement par la WTST et l'identification ROY indique que la solution a été produite par l'algorithme modifié de ROY et al. (1985). L'information audessus du tableau 3 est générale et s'applique à l'ensemble des solutions présentées. On retrouve le nombre de requêtes, le nombre de passagers, le total du temps de déplacement direct des requêtes (le temps de déplacement direct est le temps de déplacement, sans détour, de l'origine à la destination de la requête), le nombre de pièces de travail ainsi que l'amplitude totale disponible. Dans le tableau on retrouve: le nombre de tâches résultant du morcellement, le type de temps de déplacement utilisé, l'amplitude effective de la solution (i.e. amplitude disponible moins le temps libre au début et à la fin de chacune des pièces), le temps réellement travaillé par les chauffeurs excluant les temps libres sans passager, le total du temps libre des chauffeurs, la répartition des blocs de temps libre (périodes consécutives de temps libre), les ratios de performance des itinéraires évaluant le nombre de passagers transportés par heure, le nombre d'itinéraires construits ainsi que le nombre de violations des contraintes de capacité et des contraintes de fenêtres horaires. Notons que le temps libre utile correspond à la somme des heures des blocs de 3 heures et plus. On considère que les plus petits blocs sont moins favorables à l'insertion éventuelle des requêtes du jour d'opération.

Tableau 3: Description des solutions initiales

Nombre de requêtes:	1890
Nombre de passagers:	2107
Total du temps direct des requêtes:	485 heures
Nombre de pièces de travail:	133
Amplitude disponible:	1130 heures

IDENTIFICATION	TOR-5m	ROY-5m	ROY-1m
Nombre de tâches	826	638	646
Temps de déplacement	aux 5 min.	aux 5 min.	aux min.
Amplitude effective (hrs)	869	814	845
Temps travaillé (hrs)			
-Total	673	684	745
-Avec passagers	514	550	588
-Sans passager	159	134	157
Temps libre (hrs)			
-Total	457	446	385
-Utile (blocs de 3 hrs et +)	201	345	261
Blocs de Temps libre: nb. de blocs (nb. d'heures)			
blocs de 0-2 heures	732 (256)	290 (101)	336 (124)
blocs de 3-5 heures	1 (3)	41 (148)	40 (146)
blocs de 6-8 heures	0 (0)	10 (70)	4 (32)
blocs de 9-11 heures	2 (18)	8 (79)	8 (83)
blocs de 12 et + heures	15 (180)	4 (48)	0 (0)
Ratios (pass./hrs)			
Passagers/Amplitude effec.	2.4	2.6	2.5
Passagers/ Temps travaillé	3.1	3.1	2.8
Passagers/T. tr. avec pass.	4.1	3.8	3.6
Nombre d'itinéraires	116	112	116
Nombre de violations			
-de capacité	7	0	0
-de fenêtres horaires	201	0	0

La première solution initiale (TOR-5m), produite manuellement par des répartiteurs, provient de la WTST. Le mode de fonctionnement de la WTST met principalement l'accent sur la productivité maximale des véhicules; les clients réservent quelques jours d'avance, les itinéraires sont produit par la suite et le temps exact de prise est alors confirmé par téléphone. Un système informatique a été utilisé pour la manipulation d'informations telles les réservations et les confirmations ainsi que pour l'aide à la confection d'itinéraires. Le processus de confection d'itinéraires commence par un survol des requêtes qui sont regroupées par heure et principale zone géographique. Ensuite les répartiteurs utilisent le système informatique pour affecter ou déplacer les requêtes dans les itinéraires. Notons que les temps de déplacement utilisés pour confectionner les itinéraires sont évalués aux 5 minutes par les répartiteurs et que leur solution ne respecte pas les contraintes de temps à 196 reprises. Dans la solution présentée, les temps de déplacement ont été remplacés par ceux évalués par notre fonction d'évaluation des temps. Ce remplacement engendre seulement 5 violations supplémentaires des fenêtres horaires. De plus, la capacité des véhicules n'est pas respecté à 7 reprises dans la solution de la WTST.

Les deux autres solutions (ROY-5m et ROY-1m) ont été obtenues par l'algorithme modifié de Roy et al. (1985) de l'étape 1 de DARSY. Il n'y a aucune violation des contraintes de capacité et de fenêtres horaires.

Notons qu'il est difficile de comparer ces solutions. D'une part TOR-5m et ROY-5m sont construites à partir d'une fonction de temps aux 5 minutes tandis que ROY-1m à partir d'une fonction de temps aux minutes. D'autre part TOR-5m ne respecte pas toutes les contraintes tandis que les deux autres les respectent.

La description des tâches associées à chacune des solutions initiales est présentée au tableau 4. Pour chacune des solutions, le tableau présente:

- **Le nombre de tâches**. Puisque certains véhicules ne peuvent transporter de fauteuils roulants, nous distinguons le nombre de tâches impliquant au moins un fauteuil roulant ainsi que le nombre de tâches n'en impliquant pas. On remarque que les solutions ROY possèdent beaucoup moins de tâches sans fauteuil.

- **La durée des tâches.** On note une différence significative entre les solutions de ROY et la solution TOR-5m. La moyenne des durées des tâches des solutions ROY est d'environ 40 % plus élevée que celle de la solution TOR-5m. Ce qui laisse beaucoup moins de flexibilité à l'optimisation.

- **La largeur des fenêtres.** La fenêtre horaire d'une tâche est déterminée par les fenêtres horaires des arrêts de l'itinéraire de la tâche. Elle est calculée de façon à ce que l'itinéraire de la tâche demeure réalisable peu importe l'heure de service de la tâche dans sa fenêtre horaire. Les largeurs des fenêtres horaires des tâches sont par contre trop grandes pour permettre une optimisation rapide. On a alors réduit les fenêtres horaires des tâches pour obtenir des fenêtres d'une largeur maximum de 15 minutes.

- **Le nombre de requêtes par tâche.** Le nombre moyen de requêtes par tâche varie entre 2 et 3 et le nombre maximum est de 15 requêtes.

Tableau 4: Description des tâches obtenues par morcellement des solutions initiales

IDENTIFICATION	TOR-5m	ROY-5m	ROY-1m
Nombre de tâches			
-Total	826	638	617
-Avec fauteuil(s)	417	383	401
-Sans fauteuil	409	255	216
Durée des tâches			
-moyenne (min.)	35.2	50.9	57.3
-maximum (min.)	255	295	267
-écart type (min.)	30.6	43.2	50.3
Largeur des fenêtres			
-moyenne (min.)	12.8	11.3	11.5
-maximum (min.)	15	15	15
-écart type (min.)	4.3	5.4	5.1
Nb. de requêtes par tâche			
-moyenne	2.3	3.0	3.1
-maximum	12	15	14
-écart type	1.8	2.4	2.5

5.2.4 Analyse des résultats

Les tableaux 5, 6 et 7 présentent les solutions produites par DARSY basées respectivement sur les solutions initiales TOR-5m, ROY-5m et ROY-1m. Le tableau 8 présente des solutions produites avec différentes flexibilités sur les fenêtres horaires.

Le tableau 5 présente les solutions TOR-5m, A, B et C. Les comparaisons qu'on peut effectuer sont les suivantes:

- Comparaison entre les solutions produites par DARSY et la solution manuelle de la WTST.
- Comparaison entre la fonction économique de minimisation du déplacement ($\alpha = 1$) et celle de la maximisation des durées des périodes libres ($\alpha = 1.2$). .
- Comparaison entre la solution B et la solution C produite en ré-optimisant la solution B.

Les remarques locales sur le tableau 5 sont les suivantes:

- Les solutions B et C ont une amplitude effective plus petite de 12 à 13 % comparativement aux solutions TOR-5m et A.
- Les solutions A, B et C effectuent respectivement 24 %, 14 % et 11 % moins de déplacement à vide (temps travaillé sans passager) que la solution TOR-5m.
- Le temps libre total est de 7 à 9 % plus élevé pour les solutions A, B et C comparativement à la solution TOR-5m.
- Les solutions A, B et C possèdent respectivement 2 %, 76 % et 93 % plus de temps libre utile. Le nombre de blocs de temps libre de 3 heures et + est plus élevé.
- Les solutions B et C sont plus efficaces en termes des ratios des nombres

de passagers transportés par heure.

Les remarques générales sur le tableau 5 sont les suivantes:

- Des solutions très efficaces sont obtenues avec DARSY (la solution A pour son déplacement et les solutions B et C pour leurs déplacement et leurs temps libres utiles) comparativement à la solution TOR-5m.

- La fonction économique avec $\alpha = 1.2$ utilisée pour générer les solutions B et C fournie des résultats très intéressant. Tout en minimisant le temps de déplacement à vide elle contribue à regrouper les périodes de temps libre pour une meilleure insertion éventuelle des requêtes.

- La ré-optimisation permet un gain assez important dans le regroupement du temps libre sans pour autant augmenter de beaucoup le temps de déplacement.

Les figures 4 et 5 présentent respectivement les périodes de temps libres des solutions TOR-5m et C. Chaque ligne représente le travail d'un chauffeur. Les traits pleins indiquent les périodes de temps libre. Pour un survol rapide des chauffeurs ayant une proportion élevée de temps libre dans leur journée, la proportion de temps libre supérieure à 50 % est indiquée dans la colonne de gauche. On remarque rapidement la supériorité de la solution C pour l'insertion éventuelle de nouvelles requêtes.

Tableau 5: Comparaison des solutions TOR-5m, A, B et C.

IDENTIFICATION	TOR-5m	A Optimisation distance minimum	B Optimisation temps libre maximum	C Re-optimisation temps libre maximum
Amplitude effective (hrs)	869	860	762	754
Temps travaillé (hrs)				
-Total	673	631	640	640
-Avec passagers	514	510	502	499
-Sans passager	159	121	137	141
Temps libre (hrs)				
-Total	457	499	490	490
-Utile (blocs de 3 hrs et +)	201	206	353	387
Blocs de Temps libre: nb. de blocs (nb. d'heures)				
blocs de 0–2 heures	732 (256)	633 (293)	453 (137)	383 (103)
blocs de 3–5 heures	1 (3)	3 (8)	17 (60)	18 (58)
blocs de 6–8 heures	0 (0)	0 (0)	4 (26)	3 (38)
blocs de 9–11 heures	2 (18)	2 (18)	8 (75)	11 (111)
blocs de 12 et + heures	15 (180)	15 (180)	16 (192)	15 (180)
Ratios (pass./hrs)				
Passagers/Amplitude effec.	2.4	2.5	2.8	2.8
Passagers/ Temps travaillé	3.1	3.3	3.3	3.3
Passagers/T. tr. avec pass.	4.1	4.1	4.2	4.2
Nombre d'itinéraires	116	116	115	116
Nombre de violations				
-de capacité	7	7	7	7
-de fenêtres horaires	201	148	169	174

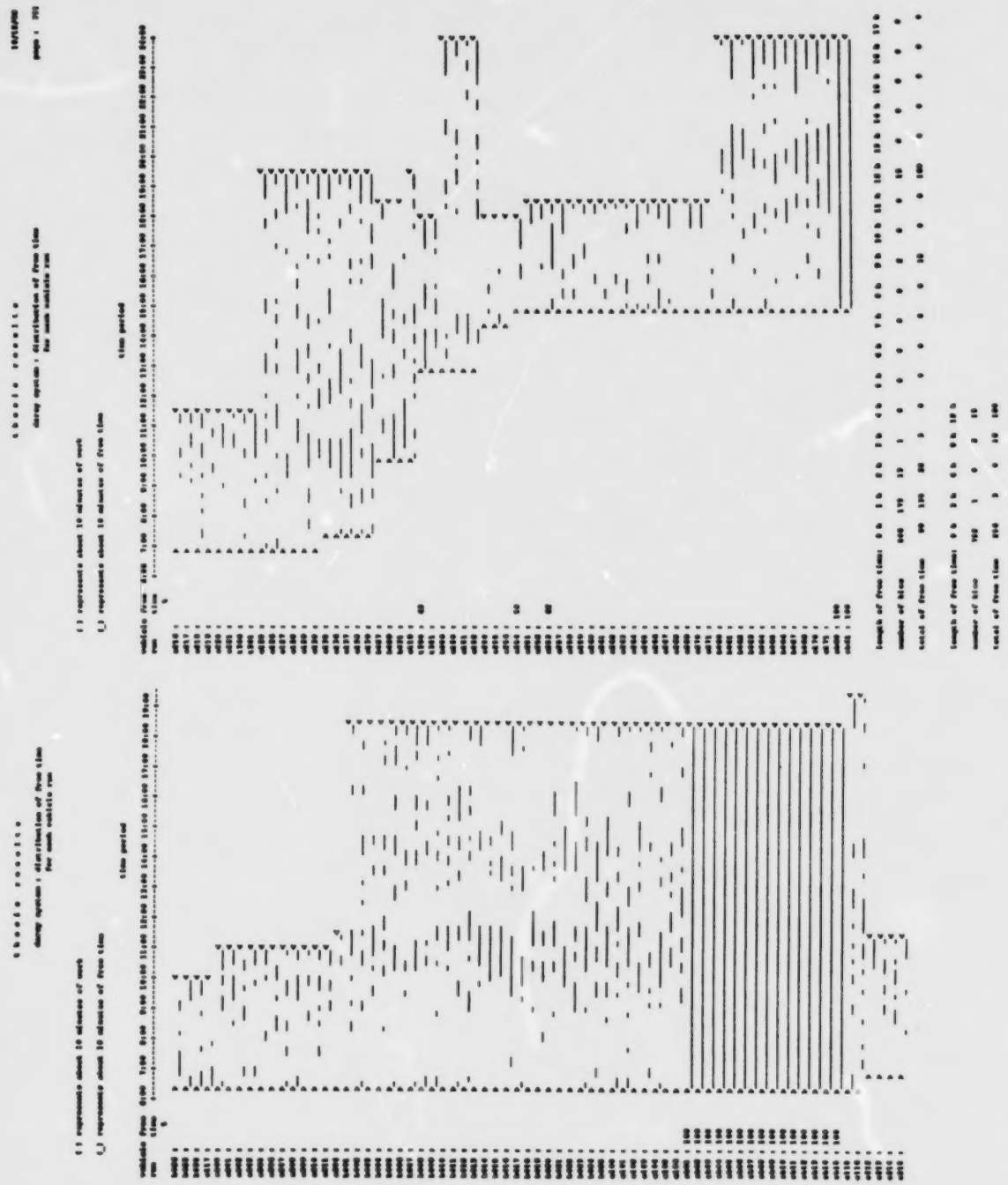


Figure 5: Répartition des périodes libres de la solution TOR-5m.

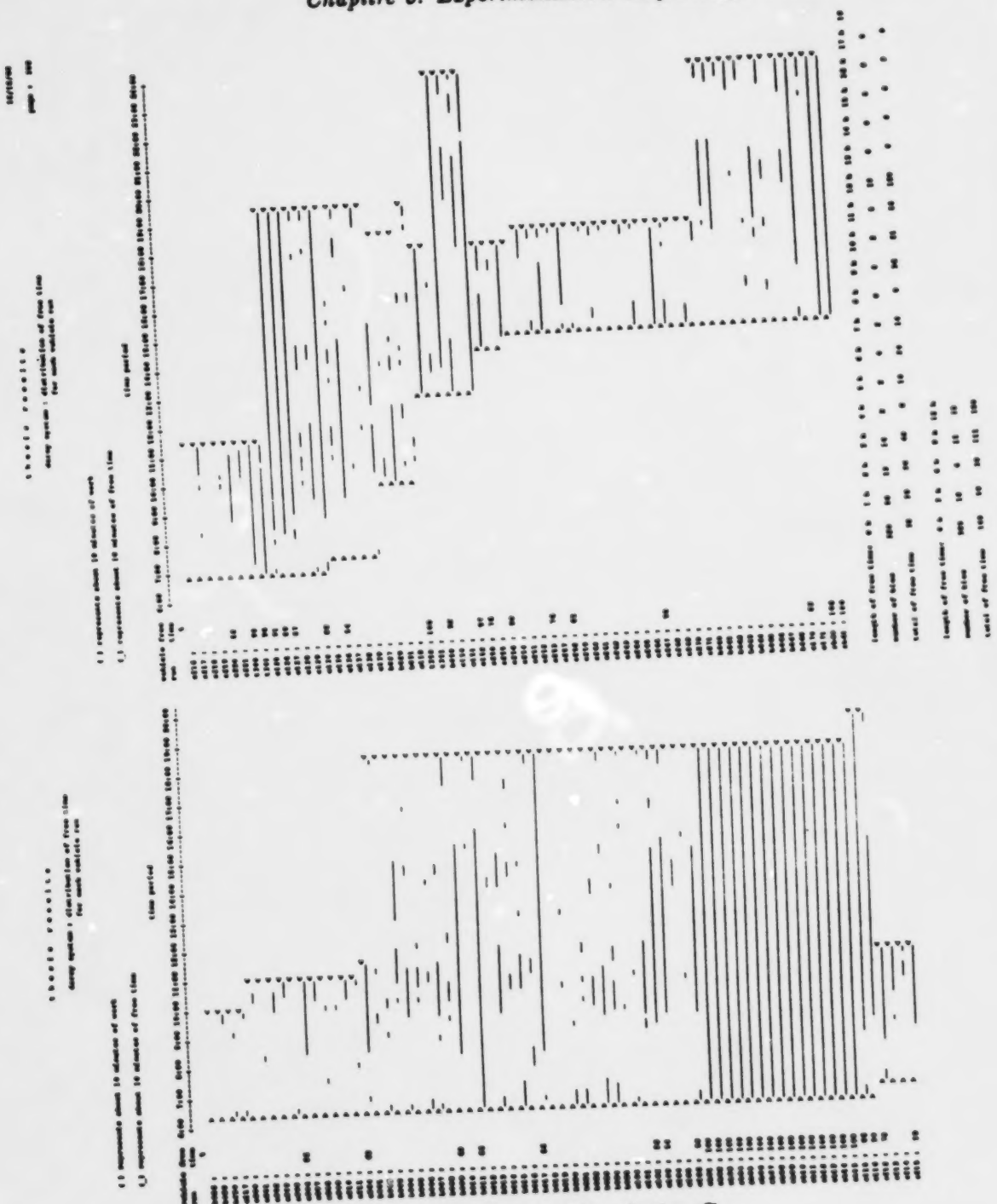


Figure 6: Répartition des périodes libres de la solution C.

Le tableau 6 présente les solutions ROY-5m, D, E et F. Notons que la solution initiale ROY-5m est morcellée en un moins grand nombre de tâches (638 tâches) comparativement à TOR-5m (826 tâches). Ceci a pour effet de fournir moins de flexibilité à l'optimisation. Les comparaisons que l'on peut effectuer sont les suivantes:

- Comparaison entre les solutions obtenues avec DARSY et la solution ROY-5m générée par l'algorithme de ROY et al. (1985).
- Comparaison entre la fonction économique de minimisation du déplacement ($\alpha = 1$) et celle de la maximisation des durées des périodes libres ($\alpha = 1.2$).
- Comparaison entre la solution E et la solution F produite en ré-optimisant la solution E.

Les remarques locales sur le tableau 6 sont les suivantes:

- Les solutions ont des amplitudes effectives du même ordre de grandeur.
- Les solutions D, E et F effectuent respectivement 29 %, 20 % et 19 % moins de déplacement à vide (temps travaillé sans passager) que la solution ROY-5m. Notons que la réduction est plus importante que lorsque la solution initiale est TOR-5m.
- Le temps libre total est de 4 à 5 % plus élevé pour les solutions D, E et F comparativement à la solution ROY-5m.
- Les solutions D, E et F possèdent respectivement -29 %, 9 % et 10 % plus de temps libre comparativement à ROY-5m. Notons que ces gains sont plus faibles que lorsque la solution initiale est TOR-5m.

Les remarques générales sur le tableau 6 sont les suivantes:

- Des solutions de qualité supérieure sont obtenues avec DARSY comparativement à l'algorithme de ROY et al. (1985).

- Notre approche dépend beaucoup de la solution initiale puisque c'est de cette solution que sont créées les tâches. Plus le nombre de tâches est petit, moins la flexibilité durant l'optimisation est grande. On remarque effectivement moins d'augmentation du temps libre utile avec la solution initiale ROY-5m (de 638 tâches) comparativement à la solution TOR-5m (de 826 tâches). Toutefois, on remarque une diminution plus grande du temps de déplacement à vide.

- La supériorité de la fonction économique maximisant la durée des périodes libres est encore plus apparente.

- La ré-optimisation n'apporte vraiment pas beaucoup.

Le tableau 7 présente les solutions ROY-1m, G, H et I. Les comparaisons que l'on peut effectuer sont sensiblement les mêmes qu'au tableau 6. La différence est au niveau de la fonction de temps de déplacement utilisée. Celle-ci génère des temps de déplacement aux minutes plutôt qu'aux 5 minutes. Les remarques locales sur le tableau 7 sont les suivantes:

- La solution G a une amplitude effective largement supérieure aux autres solutions.

- Les solutions G, H et I effectuent respectivement 38 %, 21 % et 25 % moins de déplacement à vide (temps travaillé sans passager) que la solution ROY-1m. Notons que la réduction est plus importante que dans les 2 tableaux précédents.

- Le temps libre total est de 7 à 11 % plus élevé pour les solutions G, H et I comparativement à la solution ROY-1m.

- Les solutions G, H et I possèdent respectivement -38 %, 18 % et 16 % plus de temps libre utile.

- La solution I, résultant de la ré-optimisation de la solution H, a des caractéristiques surprenantes. Comparée à la solution H, la solution I procure moins de temps libre utile (de 2 %) et moins de temps de déplacement à

vide (de 6 %).

- Moins d'itinéraires sont nécessaire dans les solutions H et I.

Les remarques générales sur le tableau 7 sont les suivantes:

- Encore une fois, DARSY produit des solutions bien supérieures à celles produitent par l'algorithme de Roy et al. (1985).

- Bien que la fonction économique minimisant le déplacement procure une économie de 38 % sur le déplacement, elle diminue également le temps libre utile du même rapport. La fonction économique maximisant les durées des périodes libres procure des résultats très satisfaisants.

- La ré-optimisation de la solution H apporte très peu.

Le tableau 8 présente les solutions G, J, K et H qui permettent d'évaluer l'effet de la variation de la flexibilité maximale sur les fenêtres horaires. Les solutions G et H ont été générées en imposant une flexibilité maximale de 15 minutes alors qu'elle est de 30 minutes pour les solutions J et K. Les solutions générées avec une plus grande flexibilité sont significativement supérieures. La solution J améliore, par rapport à la solution G, le temps de déplacement de 9 % mais par contre réduit le temps libre utile du même rapport. La solution K améliore, par rapport à la solution H, le temps de déplacement de 13 % et le temps libre utile de 4 %.

Les conclusions que l'on peut tirées de l'ensemble des optimisations sont les suivantes:

- Comparé à la solution manuelle TOR-5m, le prototype DARSY, avec l'aide de l'heuristique de ROY et al. (1985), produit une solution bien supérieure (solution E). En plus de respecter toutes les contraintes, le temps total travaillé passe de 673 h à 662 h, le temps libre utile passe de 201 h à 375 h (augmentation de 90 %) et le nombre d'itinéraires passe de 116 à 112.

- Il vaut mieux optimiser avec une fonction de temps aux minutes plutôt qu'aux 5 minutes.

- La fonction économique basée sur les durées des périodes fournit des résultats très intéressants et offre beaucoup de potentiel.

- La ré-optimisation des solutions générées par DARSY apporte très peu d'amélioration, ce qui démontre en partie la qualité du prototype DARSY.

- La variation de la flexibilité sur les fenêtres horaires permet de s'ajuster selon le désir d'obtenir soit une bonne solution rapidement ou un très bonne solution en prenant plus de temps.

Pour terminer cette section sur l'analyse des résultats voici quelques informations sur les temps de résolutions. L'étape 1 de la procédure générale qui utilise l'algorithme de Roy et al. (1985) prend près de 2 h 40 sur un SUN 3/50 avec un co-processeur mathématique. L'étape 2, ré-optimisant les itinéraires un à un, prend moins d'une seconde par itinéraire lorsque les fenêtres horaires sont réduites. Le temps pris par l'étape 3 est négligeable. L'étape 4 prend de 2 h 30 (solution H) à 7 h 30 (solution B) en temps CYBER 173 avec 35 à 50 petits problèmes m-TSPTW à résoudre. Dans l'algorithme de génération de colonnes pour résoudre les problèmes m-TSPTW, 75 à 90 % du temps d'exécution est utilisé par l'algorithme du simplexe (celui de Land et Powell, 1973). Des essais nous permettent de croire que l'utilisation d'un nouveau simplexe tel XMP (Marsten, 1981) permettrait de réduire le temps utilisé par le simplexe de 2 à 4 fois. Finalement, les temps d'exécution des optimisations basées sur les solutions initiales TOR-5m et ROY-5m sont respectivement 2.5 fois et 1.5 fois plus grand que les temps d'exécution de ceux basées sur la solution initiale ROY-1m. L'augmentation de la flexibilité de 15 à 30 minutes implique près de 1.5 fois plus de temps de résolution lorsque l'optimisation est basée sur la solution initiale ROY-1m.

Tableau 6: Comparaison des solutions ROY-5m, D, E et F.

IDENTIFICATION	ROY-5m	D Optimisation distance minimum	E Optimisation temps libre maximum	F Re-optimisation temps libre maximum
Amplitude effective (hrs)	814	874	799	787
Temps travaillé (hrs)				
-Total	684	666	662	667
-Avec passagers	550	570	555	558
-Sans passager	134	95	107	109
Temps libre (hrs)				
-Total	446	464	468	463
-Utile (blocs de 3 hrs et +)	345	244	375	380
Blocs de Temps libre: nb. de blocs (nb. d'heures)				
blocs de 0-2 heures	290 (101)	424 (221)	307 (94)	288 (82)
blocs de 3-5 heures	41 (148)	34 (116)	33 (120)	35 (129)
blocs de 6-8 heures	10 (70)	3 (18)	12 (70)	8 (49)
blocs de 9-11 heures	8 (79)	6 (62)	11 (113)	14 (142)
blocs de 12 et + heures	4 (48)	4 (48)	6 (72)	5 (60)
Ratios (pass./hrs)				
Passagers/Amplitude effec.	2.6	2.4	2.6	2.7
Passagers/ Temps travaillé	3.1	3.2	3.2	3.2
Passagers/T. tr. avec pass.	3.8	3.7	3.8	3.8
Nombre d'itinéraires	112	112	112	112
Nombre de violations				
-de capacité	0	0	0	0
-de fenêtres horaires	0	0	0	0

Tableau 7: Comparaison des solutions ROY-1m, G, H et I.

IDENTIFICATION	ROY-1m	G Optimisation distance minimum	H Optimisation temps libre maximum	I Re-optimisation temps libre maximum
Amplitude effective (hrs)	845	942	851	836
Temps travaillé (hrs)				
-Total	745	703	718	709
-Avec passagers	588	605	594	592
-Sans passager	157	98	124	117
Temps libre (hrs)				
-Total	385	427	412	421
-Utile (blocs de 3 hrs et +)	261	161	308	302
Blocs de Temps libre: nb. de blocs (nb. d'heures)				
blocs de 0–2 heures	336 (124)	492 (266)	358 (104)	361 (119)
blocs de 3–5 heures	40 (146)	42 (144)	40 (150)	37 (143)
blocs de 6–8 heures	4 (32)	1 (8)	5 (35)	4 (23)
blocs de 9–11 heures	8 (83)	1 (9)	11 (110)	12 (123)
blocs de 12 et + heures	0 (0)	0 (0)	1 (13)	1 (13)
Ratios (pass./hrs)				
Passagers/Amplitude effec.	2.5	2.2	2.5	2.5
Passagers/ Temps travaillé	2.8	3.0	2.9	3.0
Passagers/T. tr. avec pass.	3.6	3.5	3.4	3.6
Nombre d'itinéraires	116	116	114	113
Nombre de violations				
-de capacité	0	0	0	0
-de fenêtres horaires	0	0	0	0

Tableau 8: Comparaisons des solutions G, J, H et K.

IDENTIFICATION	G Optimisation distance minimum flex.: 15 m.	J Optimisation distance minimum flex.: 30 m.	H Optimisation temps libre maximum flex.: 15 m.	K optimisation temps libre maximum flex.: 30 m.
Amplitude effective (hrs)	942	937	851	837
Temps travaillé (hrs)				
-Total	703	701	718	702
-Avec passagers	605	612	594	594
-Sans passager	98	89	124	108
Temps libre (hrs)				
-Total	427	429	412	428
-Utile (blocs de 3 hrs et +)	161	146	308	317
Blocs de Temps libre: nb. de blocs (nb. d'heures)				
blocs de 0-2 heures	492 (266)	481 (283)	358 (104)	342 (111)
blocs de 3-5 heures	42 (144)	40 (137)	40 (150)	37 (140)
blocs de 6-8 heures	1 (8)	0 (0)	5 (35)	7 (41)
blocs de 9-11 heures	1 (9)	1 (9)	11 (110)	11 (112)
blocs de 12 et + heures	0 (0)	0 (0)	1 (13)	2 (24)
Ratios (pass./hrs)				
Passagers/Amplitude effec.	2.2	2.2	2.5	2.5
Passagers/ Temps travaillé	3.0	3.0	2.9	3.0
Passagers/T. tr. avec pass.	3.5	3.4	3.4	3.5
Nombre d'itinéraires	116	116	114	115
Nombre de violations				
-de capacité	0	0	0	0
-de fenêtres horaires	0	0	0	0

5.3 Références

Land, A.H. and S. Powell (1973) Fortran Codes for Mathematical Programming: Linear, Quadratic and Discrete. John Wiley & Sons eds.

Marsten, R.E. (1981) The design of the XMP Linear Programming Library. *ACM Transaction on Mathematical Software*, vol. 7, no. 4, December, 481–497.

Roy, S., J.-M. Rousseau, G. Lapalme et J.A. Ferland (1985) Élaboration des routes et horaires de véhicules pour le transport adapté: L'algorithme. Centre de recherche sur les transport, publication #412f, Université de Montréal, 21 pages.

CONCLUSION

Cette thèse a présenté un ensemble d'algorithmes permettant de confectionner les itinéraires et l'horaire de véhicules en vue du transport de personnes handicapées. Les contributions apportées sont nombreuses. Les principales sont: (1) le concept innovateur de tâche ("mini-cluster"), (2) la généralisation d'un algorithme de génération de colonnes pour le problème de voyageur de commerce multiple avec fenêtre horaire pour résoudre les problèmes avec plusieurs dépots et des véhicules de caractéristiques différentes, (3) une méthode de décomposition géographique et temporelle permettant de résoudre de très gros problèmes de voyageur de commerce multiple avec fenêtres horaires, (4) un algorithme exact de programmation dynamique permettant de résoudre de beaucoup plus gros problèmes de voyageur de commerce avec contraintes additionnelles de temps, de capacité et de priorité, (5) un algorithme dual, de complexité linéaire, pour confectionner l'horaire d'un itinéraire sous des fonctions d'inconvénient convexes sur les heures de passage et, finallement, (6) une procédure originale d'estimation des temps de déplacement.

Nos objectifs qui étaient de proposer une approche de résolution pouvant traiter des problèmes de transport adapté de grande taille et permettant de générer des solutions de grande qualité ont été atteints. L'expérimentation du prototype DARSY de même que l'intérêt porté par certains organismes pour les travaux décrits dans cette thèse le confirme. En collaboration avec une entreprise privée, un financement important a été accordé par le Conseil National de la Recherche du Canada (CNRC) et par le Conseil de Recherches en Sciences Naturelles et en Génie du Canada (CRSNG) dans le but de réaliser une version commerciale du logiciel DARSY.

Avant la mise en marché du logiciel DARSY, plusieurs directions de recherche seront considérées, notament:

(1) **La confection de tâches**: présentement, afin d'obtenir un ensemble de tâches, une solution initiale est morcellée aux endroits où le véhicule se déplace sans passager. Récemment, nous avons analysé et fait analyser par la WTST les tâches produites par le morcellement des solutions initiales

engendrées par l'algorithme de ROY. Les résultats des analyses nous ont indiqué que les tâches étaient très peu directionelles et en moyenne beaucoup trop longues, d'où la nécessité d'une nouvelle approche de confection de tâches. Il est beaucoup plus facile de bien regrouper des requêtes voisines plutôt que d'affecter les requêtes aux véhicules. Nous travaillons présentement à l'élaboration de deux algorithmes, l'un étant heuristique et l'autre étant une approche exacte pour la confection de tâches. L'approche exacte sera basée sur la méthode de génération de colonnnes. Les caractéristiques des tâches qui seront recherchées sont les suivantes: (a) la directionnalité: les itinéraires des tâches doivent conserver une certaine direction pour ne pas donner aux usagers l'impression que le véhicule se déplace inéfficacement, (b) la faisabilité: la portion d'itinéraire de chaque tâche doit satisfaire l'ensemble des requêtes de la tâche en respectant les contraintes de fenêtres de temps, de couplage, de priorité et de capacité du véhicule, (c) l'efficacité: la somme des temps de déplacement des véhicules à l'intérieur des tâches doit être assez proche du minimum, (d) la qualité du service: la qualité de service des usagers doit être acceptable.

(2) **Les pauses et les repas des chauffeurs**: jusqu'à présent nous avons considéré que les chauffeurs peuvent travailler sans interruption pendant leur journée de travail. La construction d'itinéraires efficaces engendre beaucoup moins de périodes libres pour ces chauffeurs. Il devient alors important d'incorporer des pauses et des périodes de repas dans les itinéraires.

(3) **Le temps d'attente**: lors de la confection d'horaire, il est important de limiter le temps d'attente lorsque des passagers sont à bord du véhicule. Il est possible de généraliser les résultats du chapitre 4 afin de borner le temps d'attente.

(4) **L'analyse de scénarios**: la mise au point de DARSY permet l'analyse de différents scénarios afin d'évaluer par exemple, l'effet de la qualité de service accordée aux usagers, l'effet de la composition de la flotte de véhicules, l'effet des types de pièces de travail des chauffeurs, l'effet des

modes d'opération des société de transport, etc.

Ces travaux de recherche permettrons de finaliser la conception du logiciel DARSY. Les premier tests, en site pilote, sont prévus pour le printemps 1990.

A

Estimation des temps de déplacement

A.1. Introduction

Les tests numériques nécessitent le développement d'une fonction d'estimation des temps de déplacement. Cette fonction doit fournir des temps, aux 5 minutes, comparables à ceux évalués par les répartiteurs. La fonction de temps à estimer est donc discrète. Sans s'aventurer dans la théorie formelle de la statistique, nous présentons, dans le cadre de notre application, quelques modèles de régression discrète, la formulation des problèmes d'optimisation pour estimer les paramètres ainsi qu'un algorithme pouvant résoudre ces problèmes d'optimisation.

Pour éviter la confusion, nous faisons une distinction entre le "temps de déplacement" et le "temps d'inter-arrêts". Le premier correspond aux temps de déplacement du véhicule tandis que le second inclut le temps de déplacement et le temps de service à l'arrêt d'origine. Ainsi, nous avons 3028 observations de temps d'inter-arrêts dont la description est donnée à la section A.2. Les modèles de régression sont présentés aux sections A.3, A.4 et A.5, les problèmes d'optimisation à la section A.3.4, les démonstrations mathématiques des propositions à l'appendice A.A et l'algorithme de résolution des problèmes d'optimisation à l'appendice A.B.

A.2. Description des données

A partir de l'ensemble des itinéraires confectionnés à la Wheel Trans Service de Toronto pour une journée d'opération donnée, nous avons extrait l'information relative aux déplacements des véhicules entre chacun des arrêts, et exclu les déplacements nuls entre les arrêts consécutifs ayant la même adresse. Cette information nous a permis de recueillir 3028 observations comprenant pour chacune d'elles:

1) l'adresse de départ,

2) l'adresse d'arrivée,

3) l'heure à laquelle le service a débuté à l'adresse de départ,

4) le temps d'inter-arrêts évalué aux 5 minutes,

5) le temps de service des passagers à l'adresse de départ (en minutes).

Les principaux facteurs identifiés pouvant influencer l'évaluation du temps d'inter-arrêts sont: la distance à parcourir, la région du déplacement, le temps de service à l'adresse de départ et la période de la journée.

La région du déplacement est considérée implicitement en utilisant des distances corrigées, i.e. des distances pondérées selon les différentes zones traversées. L'évaluation de ces distances entre deux adresses est obtenue à partir d'une matrice de distance corrigée entre les différentes zones (géocodées) du Toronto métropolitain. La dimension de la matrice est de 1064 par 1064 zones où chacune des zones couvre 1/4 mille carré. La matrice est construite à partir d'un graphe qui représente le réseau routier. Dans le graphe, un arc représente un segment de rue alors qu'un noeud est représenté par une intersection. A chaque arc est associée la distance du segment de rue qu'il représente et celle-ci est pondérée selon le degré de congestion de la zone du segment de rue. Chaque zone possède un noeud centroïde correspondant à son centre ou à son plus grand point d'intérêt. Ainsi, les distances corrigées de la matrice sont évaluées en déterminant les plus courts chemins dans le graphe entre les centroïdes des zones sauf pour les éléments sur la diagonale où la distance est évaluée par un déplacement moyen dans la zone. Finalement, la distance corrigée entre deux adresses est celle fournie par la matrice entre les zones correspondantes, si les adresses sont distinctes et la distance est nulle sinon.

Pour terminer cette section, nous présentons trois figures. La figure 1 illustre les temps d'inter-arrêts évalués aux 5 minutes en fonction des distances corrigées. On observe une grande variation des temps pour une distance donnée. Nous verrons par la suite que cette variation s'explique en grande partie par le temps de service et la période de la journée du déplacement. Les figures 2 et 3 présentent respectivement, la répartition des temps d'inter-arrêts et des temps de service à l'adresse de départ. De la figure 2, on constate que 96% des temps d'inter-arrêts sont inférieurs ou égaux à 20 minutes et que 39% sont de 10 minutes. La figure 3, nous indique que la majorité des temps de service sont de 1 et 3

minutes. Selon la formule du calcul, ces temps de service sont respectivement alloués pour l'aide nécessaire à une personne sans fauteuil roulant et à une personne avec fauteuil roulant.

A.3. Modèle de régression unidimensionnel

Nous considérons un premier modèle de régression discrète où la seule variable indépendante est la distance corrigée et où la fonction de régression est discrète.

Soient T_k le temps d'inter-arrêts de la k-ième observation, $\beta_1, \ldots, \beta_{p-1}$ les paramètres découpant le domaine de la fonction en p intervalles, D_k la variable représentant la distance corrigée et ϵ_k l'erreur aléatoire du modèle. Le modèle $(M1)$ s'écrit alors comme suit:

$$(M1) \qquad T_k = \epsilon_k + \begin{cases} 5 & \text{si} \quad 0 < D_k \leq \beta_1 \\ 10 & \text{si} \quad \beta_1 < D_k \leq \beta_2 \\ \vdots & \quad \vdots \\ 5p & \text{si} \quad \beta_{p-1} < D_k. \end{cases}$$

Le modèle divise le domaine des valeurs de la distance en p intervalles et à chaque intervalle correspond un temps d'inter-arrêts multiple de 5 minutes. La fonction de régression prend la forme d'un escalier où la hauteur des marches est de 5 minutes et leurs longueurs sont déterminées par les intervalles entre les β.

Notons par $\hat{\beta}_j$ la valeur estimée de β_j avec $\hat{\beta}_0 = 0$ et $\hat{\beta}_p = \infty$. On obtient alors la fonction suivante pour prédire le temps d'inter-arrêts en fonction de la distance corrigée:

$$T(D) = \begin{cases} 0 & \text{si} \quad D = 0 \\ 5j & \text{si} \quad \hat{\beta}_{j-1} < D \leq \hat{\beta}_j \quad j \in \{1, \ldots, p\}. \end{cases} \qquad (2)$$

Nous présentons maintenant trois méthodess pour estimer les paramètres: la méthodes des moindres carrés, celle de la somme des erreurs absolues et celle du nombre minimum d'erreurs. Suivra ensuite la section A.3.4 sur les problèmes d'optimisation résultant de l'estimation des paramètres. Les méthodess d'estimation sont présentées en utilisant la notation supplémentaire suivante: $e_k = T_k - \hat{T}_k$ est l'erreur d'estimation associée à la k-ième observation alors que $g_j(D)$ est la fonction donnant le nombre d'observations ayant un temps d'inter-arrêts de $5j$ minutes et effectuant un déplacement d'une distance d'au plus D, $j = 1, \ldots, p$.

A.3.1. La méthode des moindres carrés

Nous voulons déterminer $\beta_1, \ldots, \beta_{p-1}$ ($\beta_0 = 0$ et $\beta_p = \infty$) afin que la quantité $\sum_k (e_k)^2$ soit minimum, i.e. que la somme du carré des erreurs soit minimum. L'estimation des paramètres est basée sur l'expression de l'objectif en fonction des paramètres β_i. Le résultat suivant est démontré à l'appendice A.A.

Proposition 1: Minimiser $\sum_k (e_k)^2$ est équivalent à minimiser

$$\sum_{i=1}^{p-1} \sum_{j=1}^{p} (2j - 2i - 1) \cdot g_j(\beta_i). \tag{3}$$

On constate que la fonction à minimiser (3) est séparable. Si on représente par $\tau_i(D) = \sum_{j=1}^{p} (2j - 2i - 1) \cdot g_j(D)$ la fonction unidimensionnelle associée à l'évaluation du paramètre β_i alors la figure 4 présente l'allure des fonctions observées $\tau_1, \ldots, \tau_5$. Les courbes τ_1, τ_2 et τ_3 semblent présenter un minimum global unique, tandis que les deux autres courbes présentent des plateaux avec une infinité d'optimum globaux. Remarquons l'unimodalité des fonctions observées.

A.3.2 La méthode de la somme des erreurs absolues

Nous voulons déterminer $\beta_1, \ldots, \beta_{p-1}$ ($\beta_0 = 0$ et $\beta_p = \infty$) afin que la quantité $\sum_k |e_k|$ soit minimum, i.e. que la somme des valeurs absolues des erreurs soit minimum. L'estimation des paramètres est basée sur l'expression de l'objectif en fonction des paramètres β_i. Le résultat suivant est démontré à l'appendice A.A.

Proposition 2: Minimiser $\sum_k |e_k|$ est équivalent à minimiser

$$\sum_{i=1}^{p-1} \Big[\sum_{j=i+1}^{p} g_j(\beta_i) - \sum_{j=1}^{i} g_j(\beta_i) \Big]. \tag{4}$$

La fonction à minimiser (4) est une fois de plus séparable. Si on représente par $\phi_i(D) = \sum_{j=i+1}^{p} g_j(D) - \sum_{j=1}^{i} g_j(D)$ la fonction unidimensionnelle associée à l'évaluation du paramètre β_i, alors la figure 5 illustre l'allure des fonctions observées $\phi_1, \ldots, \phi_5$. Les courbes obtenues sont semblables à celles obtenues par la méthodes des moindres carrées. Remarquons de nouveau l'unimodalité des fonctions observées.

A.3.3. La méthode du nombre minimum d'erreurs

Nous voulons déterminer $\beta_1, \ldots, \beta_{p-1}$ ($\beta_0 = 0$ et $\beta_p = \infty$) afin que la quantité $\sum_k \min(1, |e_k|)$ soit minimum, i.e. que le nombre d'erreurs soit minimum. Cette méthodes n'est pas sensible aux données aberrantes. L'estimation des paramètres est basée sur l'expression de l'objectif en fonction des paramètres β_i. Le résultat suivant est démontré à l'appendice A.A.

Proposition 3: Minimiser $\sum_k \min(1, |e_k|)$ est équivalent à minimiser

$$\sum_{i=1}^{p-1}\Big[g_{i+1}(\beta_i) - g_i(\beta_i)\Big]. \tag{5}$$

La fonction à minimiser (5) est encore une fois séparable. Si on représente par $\sigma_i(D) = g_{i+1}(D) - g_i(D)$ la fonction unidimensionnelle associée à l'évaluation du paramètre β_i, alors la figure 6 illustre l'allure des fonctions observées $\sigma_1, \ldots, \sigma_5$. Les courbes obtenues sont très différentes des précédentes. Par exemple, les plateaux des optimums globaux sont beaucoup plus courts. De nouveau, remarquons l'unimodalité des fonctions observées.

A.3.4 Problèmes d'optimisation

Dans les sections précédentes, nous avons présenté différents objectifs à minimiser pour estimer les paramètres. Pour demeurer compatible avec le modèle $(M1)$, ces minimisations doivent se faire en considérant les contraintes $\beta_{i-1} < \beta_i$, $i = 1, \ldots, p$, provenant du modèle. Ce qui nous amène à résoudre le problème d'optimisation $(P1)$ suivant:

$$(P1) \qquad \min f(\beta_1, \ldots, \beta_{p-1}) \tag{6}$$

$$\text{sujet à} \quad \beta_{j-1} < \beta_j \qquad j = 1, \ldots, p \tag{7}$$

où f est une fonction non-linéaire propre à chaque méthode. Par contre, la fonction f s'écrit dans les trois cas, comme une somme de fonctions non-linéaires unidimensionnelles.

Les figures 4, 5 et 6 illustrent l'allure des fonctions de chacune des méthodes. On constate que les fonctions sont unimodales et que les contraintes $\beta_{i-1} < \beta_i$, $i = 1, \ldots, p$ ne sont pas réellement nécessaires. Ainsi, on peut résoudre les problèmes en minimisant séparément chaque fonction unidimensionnelle par la méthode de Fibonacci.

Pour chacune des méthodes, nous avons estimé les paramètres du modèle puis nous avons comparé les temps obtenus par notre fonction de prédiction avec

les temps observés. Le tableau 1A fournit l'estimation des paramètres obtenues par les trois méthodes. Seul $\hat{\beta}_3$ varie d'une méthode à l'autre. Le tableau 1B fournit la distribution des erreurs $T_k - \widehat{T}(D_k)$ où la fonction $\widehat{T}(D)$ est définie avec les estimateurs correspondants du tableau 1A. On remarque que les trois méthodes fournissent sensiblement les mêmes résultats statistiques. On retrouve que 67 % des valeurs sont bien estimées, 31.8 % diffèrent de 5 minutes et 0.7 % diffèrent de plus de 5 minutes. D'autre part, nous avons calculé que les proportions des temps d'inter-arrêts bien prédits sont les suivantes: 63 % sont bien prédits pour des temps observés de 5 minutes, 80.7 % pour 10 minutes, 55 % pour 15 minutes, 67 % pour 20 minutes, 36.2 % pour 25 minutes et 20 % pour 30 minutes et plus.

Pour expliquer la mauvaise qualité des estimateurs $\hat{\beta}_6$ et $\hat{\beta}_7$, analysons les valeurs obtenues. Puisque le modèle fournit un temps de $5i$ minutes lorsque la distance est dans l'intervalle $(\beta_{i-1}, \beta_i]$, la vitesse varie de $\frac{\beta_{i-1}}{5i}$ à $\frac{\beta_i}{5i}$. Ces intervalles, pour $i = 1, \ldots, 7$, calculés à partir du tableau 1A, sont illustrés à la figure 7.

On remarque que la vitesse moyenne ne croît pas avec le temps à parcourir; ce qui n'arrive pas en pratique. La vitesse est donc sous-estimée pour des grands temps (30 min. et plus). Ceci nous suggère de remplacer les contraintes $\beta_{i-1} < \beta_i$ pour $i = 1, \ldots, p$ par $\beta_{i-1} \leq \frac{1}{2} \cdot \left(\frac{i-1}{i} + \frac{i}{i+1}\right) \cdot \beta_i$ qui est un compromis entre les contraintes de vitesse minimum croissante $\beta_{i-1} \leq \frac{i-1}{i} \cdot \beta_i$ et les contraintes de vitesse maximum croissante $\beta_{i-1} \leq \frac{i}{i+1} \cdot \beta_i$.

On se retrouve donc à devoir résoudre le problème d'optimisation $(P2)$ suivant:

$$(P2) \qquad \min f(\beta_1, \ldots, \beta_{p-1}) \tag{6}$$

$$\text{sujet à} \quad \beta_{i-1} \leq \tfrac{1}{2} \cdot \left(\tfrac{i-1}{i} + \tfrac{i}{i+1}\right) \cdot \beta_i \qquad i = 1, \ldots, p. \tag{8}$$

Lorsque la fonction f est constituée d'une somme de fonctions unidimensionnellles convexes, ou encore convexes sur des intervalles, on peut résoudre ce type de problème en adaptant la théorie et l'algorithme présentés au chapitre 4. Cette adaptation est présentée à l'appendice A.B.

La nouvelle estimation des paramètres, obtenue en résolvant le problème $(P2)$, est présenté au tableau 2A. Les trois méthodes produisent la même estimation. Par rapport aux valeurs obtenues précédemment, seuls $\hat{\beta}_6$ et $\hat{\beta}_7$ sont modifiés. Les nouvelles valeurs obtenues sont maintenant plus réalistes. Finalement, on retrouve au tableau 2B la distribution des erreurs $T_k - \widehat{T}(D_k)$ où la fonction $\widehat{T}(D)$ est définie avec les estimateurs correspondants du tableau 2A. Les résultats sont semblables à ceux obtenus précédemment.

A.4. Modèle de régression bidimensionnel

Nous considérons un deuxième modèle de régression discrète où les variables indépendantes sont la distance corrigée et le temps de service à l'adresse de départ.

Soient T_k le temps d'inter-arrêts de la k-ième observation, $\beta_1, \ldots, \beta_{p-1}$ les paramètres découpant le domaine de la fonction en p région, D_k la distance corrigée, M_k le temps de service à l'adresse de départ et ϵ_k l'erreur aléatoire du modèle.

Lorsque le temps de service d'une observation est supérieur ou égal à 5 minutes, on peut réduire d'un multiple de 5 les temps d'inter-arrêts et de service sans perdre d'information. La transformation suivante est ainsi utilisée:

$$T'_k = T_k - \lfloor M_k/5 \rfloor \cdot 5 \quad \forall k \tag{9}$$

$$M'_k = M_k - \lfloor M_k/5 \rfloor \cdot 5 \quad \forall k \tag{10}$$

où $\lfloor x \rfloor$ représente le plus grand entier plus petit ou égal à x. Par exemple, si $T_k = 25$ et $M_k = 7$ alors $T'_k = 20$ et $M'_k = 2$. Remarquons que les nouveaux temps de service sont tels que $M'_k \in \{0, 1, 2, 3, 4\}$.

Le modèle $(M2)$ proposé dans cette section est une extension du modèle $(M1)$. Dans ce dernier, il est possible de remplacer β_i par $\left(\frac{\beta_i}{5i}\right) \cdot 5i$. Les $5i$ se simplifient et le modèle demeure le même. Par contre, sous cette forme, nous mettons en évidence la vitesse maximum $\left(\frac{\beta_i}{5i}\right)$ et le temps d'inter-arrêts $(5i)$. Dans

le modèle $(M2)$, on tient compte du temps de service M'_k à l'adresse de départ. On multiplie donc la vitesse maximum $\left(\frac{\beta_i}{5i}\right)$ par le temps réel de déplacement $(5i - M'_k)$. On obtient alors le modèle bidimensionnel suivant:

$$(M2) \qquad T'_k = \epsilon_k + \begin{cases} 5 & \text{si} \quad 0 < D_k \leq \frac{\beta_1}{5} \cdot (5 - M'_k) \\ 10 & \text{si} \quad \frac{\beta_1}{5} \cdot (5 - M'_k) < D_k \leq \frac{\beta_2}{10} \cdot (10 - M'_k) \\ \vdots & \vdots \\ 5p & \text{si} \quad \frac{\beta_{p-1}}{5p-1} \cdot (5p - 1 - M'_k) < D_k \leq \frac{\beta_p}{5p} \cdot (5p - M'_k). \end{cases}$$

Le modèle divise le domaine des valeurs de la distance en p régions et à chaque région correspond un temps d'inter-arrêts multiple de 5 minutes. Pour chaque valeur possible de $M'_k \in \{0, 1, 2, 3, 4\}$, la fonction de régression prend la forme d'un escalier en fonction de la distance corrigée. Notons que si $M'_k = 0 \quad \forall k$ alors le modèle $(M2)$ prend la même forme que le modèle $(M1)$. La transformation représentée par les relations (9) et (10) est décentrée et engendre un léger biais au modèle. Puisque $M'_k \in \{0, 1, 2, 3, 4\}$ le temps d'inter-arrêts de $5i$ du modèle représente les temps de déplacement réel de l'intervalle $(5(i - 1), 5i]$. Pour centrer le temps d'inter-arrêts de $5i$ sur l'intervalle $[5i-2, 5i+2]$, on utilise la transformation suivante:

$$T'_k = T_k - [M_k/5]_5 \quad \forall k \tag{12}$$

$$M'_k = M_k - [M_k/5]_5 \quad \forall k \tag{13}$$

où $[x]_5$ représente le plus proche multiple de 5 de x. Par contre, cette transformation peut engendrer des temps nul. Par exemple, si $T_k = 5$ et $M_k = 4$ alors $T'_k = 0$ et $M_k = -1$. Dans l'évaluation des valeurs des paramètres on ne considère pas les observations pour lesquelles $T'_k = 0$.

Si $\hat{\beta}_i$ représente la valeur estimée de β_i, alors on obtient la fonction suivante pour prédire le temps d'inter-arrêts en fonction de la distance D à parcourir et du temps de service M où $M' = M - [M/5]_5 \cdot 5$:

$$\widehat{T}(D, M) = [M/5]_5 \cdot 5 + \begin{cases} 0 & \text{si} \quad D = 0 \\ 5 \cdot i & \text{si} \end{cases} \tag{12}$$

$$(\frac{\hat{\beta}_{i-1}}{5(i-1)}) \cdot (5(i-1) - M') < \ D \ \leq (\frac{\hat{\beta}_i}{5i}) \cdot (5i - M') \quad i \in \{1, \ldots, p\}.$$

Soit $h_i(x, m)$ la fonction donnant le nombre d'observations ayant un temps d'inter-arrêts de $5i$ avec un temps de service m et effectuant un déplacement d'une distance d'au plus $x \cdot \frac{(5i-m)}{5i}$. Pour les trois méthodes présentées à la section précédente, l'estimation des paramètres est basée sur l'expression des objectifs en fonction des paramètres β_i. Les résultats suivants sont démontrés à l'appendice A.A.

Proposition 4: Pour le modèle bidimensionnel: minimiser $\sum_k (e_k)^2$ est équivalent à minimiser

$$\sum_{i=1}^{p-1} \sum_{j=1}^{p} \sum_{M=0}^{4} (2j - 2i - 1) \cdot h_j(\beta_i, M). \tag{13}$$

Proposition 5: Pour le modèle bidimensionnel, minimiser $\sum_k |e_k|$ est équivalent à minimiser

$$\sum_{i=1}^{p-1} \sum_{M=0}^{4} \left[\sum_{j=i+1}^{p} h_j(\beta_i, M) - \sum_{j=1}^{i} h_j(\beta_i, M) \right]. \tag{14}$$

Proposition 6: Pour le modèle bidimensionnel, minimiser $\sum_k \min(1, |e_k|)$ est équivalent à minimiser

$$\sum_{i=1}^{p-1} \left[h_{i+1}(\beta_i, M) - h_j(\beta_i, M) \right]. \tag{15}$$

Pour minimiser ces fonctions, on résout le problème d'optimisation $(P2)$ de la section précédente. Les résultats obtenus sont présentés aux tableaux 3A et 3B. Notons que les 11 observations aberrantes pour lesquelles le temps de service était supérieur ou égal au temps d'inter-arrêts $(M_k \geq T_k)$ ont été éliminées. Dans le tableau 3B, on remarque que 77.5 % des valeurs sont bien estimées, 21.7 % diffèrent de 5 minutes et seulement 0.4 % diffèrent de plus de 5 minutes. De

plus, on remarque une asymétrie dans la distribution des erreurs. Cette asymétrie indique une sous-estimation des temps d'inter-arrêts. La sous-estimation provient des observations en périodes de pointe qui sont sous-estimées. Ceci, nous amène à améliorer notre modèle à la section suivante. D'autre part, nous avons calculé que les proportions des temps d'inter-arrêts bien prédits sont les suivantes: 84 % pour les temps observés de 5 minutes, 84 % pour 10 minutes, 72 % pour 15 minutes, 45 % pour 20 minutes, 34 % pour 25 minutes et 0 % pour 30 minutes et plus.

A.5. Modèle de régression bidimensionnel avec périodes de pointe

Le dernier facteur qui nous semble important à considérer est la période de la journée. Dans un premier temps, nous déterminons l'effet de ce facteur sur les temps d'inter-arrêts. A cette fin, on applique le modèle bidimensionnel présenté à la section précédente aux observations de chaque période d'une heure de la journée. Ainsi, pour chacune de ces périodes, on obtient une estimation des paramètres β. De ces estimations, on détermine un ratio moyen permettant d'évaluer l'effet de la période de la journée sur l'estimation des paramètres. Pour définir ce ratio, on a besoin de la définition suivante: pour la période ℓ, soit $\hat{\beta}_{i,\ell}$ l'estimateur de β_i. Les proportions des observations en fonction des temps d'inter-arrêts sont respectivement, pour 5, 10, 15, 20 et 25 minutes, de 13.1 %, 39.3 %, 30.3 %, 13.2 % et de 3.7 % (voir figure 2). Les estimateurs de β du tableau 3A obtenus sur toutes les observations sont dans l'ordre: 2.4, 6.0, 11.7, 18.0 et 22.0 pour les 5 premiers paramètres; les autres sont de moindre importance. On définit alors le ratio moyen (pondéré) pour la période ℓ comme suit:

$$\left(\frac{\hat{\beta}_{1,\ell}}{2.4}\cdot(0.13)+\frac{\hat{\beta}_{2,\ell}}{6.0}\cdot(0.39)+\frac{\hat{\beta}_{3,\ell}}{11.7}\cdot(0.30)+\frac{\hat{\beta}_{4,\ell}}{18.0}\cdot(0.13)+\frac{\hat{\beta}_{5,\ell}}{22.0}\cdot(0.04)\right)$$

La figure 8 illustre les ratios moyens de chacune des périodes considérées. On distingue clairement deux périodes de pointe. On évalue que celle du matin s'étend de 7:00 à 8:45 et celle du soir de 15:30 à 17:45.

Dans une deuxième étape, on divise les observations en deux groupes: l'un pour les observations en périodes de pointe et l'autre pour les observations en périodes hors pointe. On estime ensuite, pour chacun des groupes, les paramètres du modèle bidimensionnel $(M2)$ de la section précédente. Les valeurs des paramètres estimés sont présentées au tableau 4A et la distribution des erreurs $T_k - \widehat{T}(D_k, M_k)$ est présentée au tableau 4B. La fonction $T(D, M)$ est définie avec les estimateurs de la méthode des moindres carrés du tableau 4A. Les estimations obtenues en périodes hors pointe sont presqu'identique (sauf pour $\hat{\beta}_2$ et $\hat{\beta}_3$) aux estimations obtenues, sur toutes les observations, aux tableau 3A. Par contre, les estimateurs obtenus en périodes de pointe sont beaucoup plus petits, reflétant des temps d'inter-arrêts plus long en périodes de pointe. On obtient du tableau 4B que 83 % des valeurs sont bien estimées, 16 % diffèrent de 5 minutes et seulement 0.5 % diffèrent de plus de 5 minutes. On remarque cette fois-ci une légère asymétrie dans les résultats du tableau 4B et que cette asymétrie est encore engendrée par les périodes de pointe. Notons maintenant qu'il y a une sur-estimation des temps des observations. Ce phénomène peut s'expliquer par le fait que la périodes de pointe est appliquée à toutes les zones de la ville et que les répartiteurs qui ont évalué les temps des observations ont probablement tenu compte qu'en dehors du centre ville, il n'y a pas réellement d'heures de pointe. Ce léger biais est acceptable car il est préférable et plus sécuritaire de sur-estimer les temps de déplacement dans un modèle de planification d'itinéraires que de les sous-estimer.

Finalement, nous avons calculé que: (a) les proportions des temps d'inter-arrêts bien prédits en périodes hors pointe sont les suivantes: 90 % pour les temps observés de 5 minutes, 90 % pour 10 minutes, 90 % pour 15 minutes, 78 % pour 20 minutes, 69 % pour 25 minutes et 0 % pour 30 minutes et plus; (b) les proportions des temps d'inter-arrêts bien prédits en périodes de pointe sont les suivantes: 61 % sont bien prédits pour les temps observés de 5 minutes, 87 % pour 10 minutes, 70 % pour 15 minutes, 86 % pour 20 minutes, 62 % pour 25 minutes et 40 % pour 30 minutes et plus. Notons qu'il y a très peu d'observations de 30 minutes et plus.

A.6 Conclusion

Nous avons présenté deux modèles de régression discrète permettant d'établir des fonctions de prédiction des temps d'inter-arrêts entre deux adresses d'un réseau urbain. L'estimation des paramètres de ces modèles nous a amené à formuler des problèmes non-linéaires particuliers et à proposer des approches simples de résolution. Les problèmes non-linéaires formulés ont tous des objectifs séparables en des fonctions unidimensionnelles ainsi que des contraintes liantes entre les variables. De plus, dans notre application, les fonctions sont unimodales.

En considérant les facteurs d'influence tels la distance, la région du déplacement, le temps de service et la période de la journée, nous avons élaboré une fonction de prédiction permettant d'estimer de façon juste les valeurs observées dans une proportion de 83 % et d'engendrer moins de 0.5 % d'erreurs de plus de 5 minutes. Ce résultat nous permet d'entreprendre avec confiance la comparaison équitable de notre système informatisé de confection d'itinéraires avec le système manuel de la Wheel-Trans Service de Toronto.

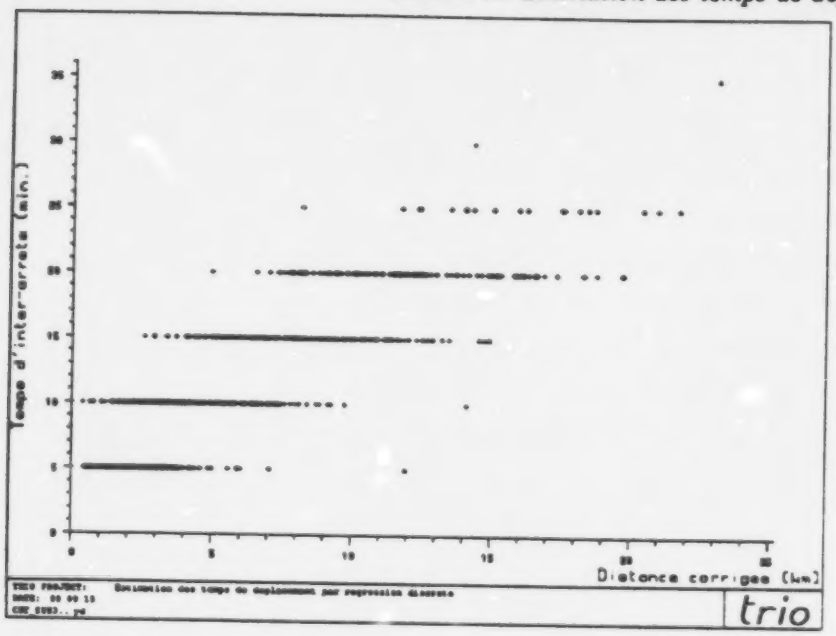


Figure 1: Distribution des temps d'inter-arrêts en fonction des distances corrigées

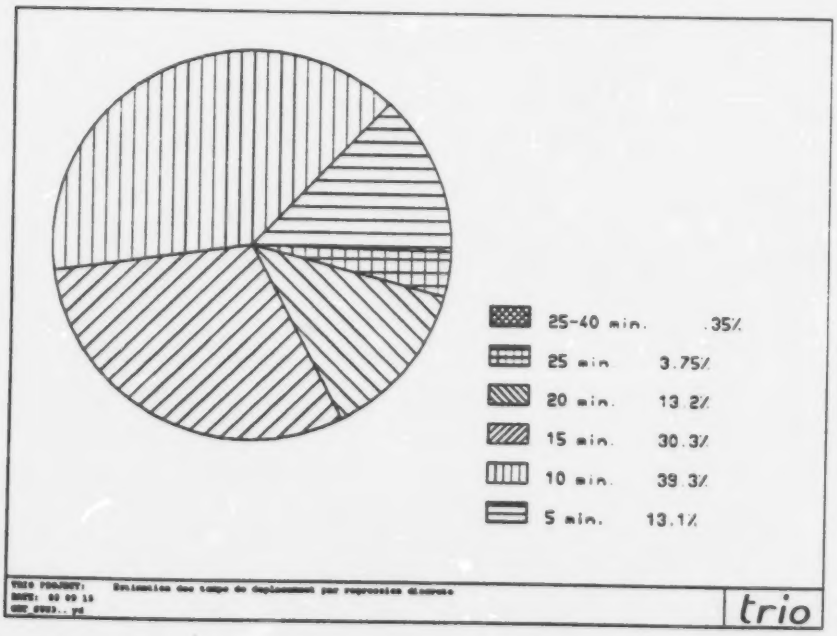


Figure 2: Répartition des temps d'inter-arrêts

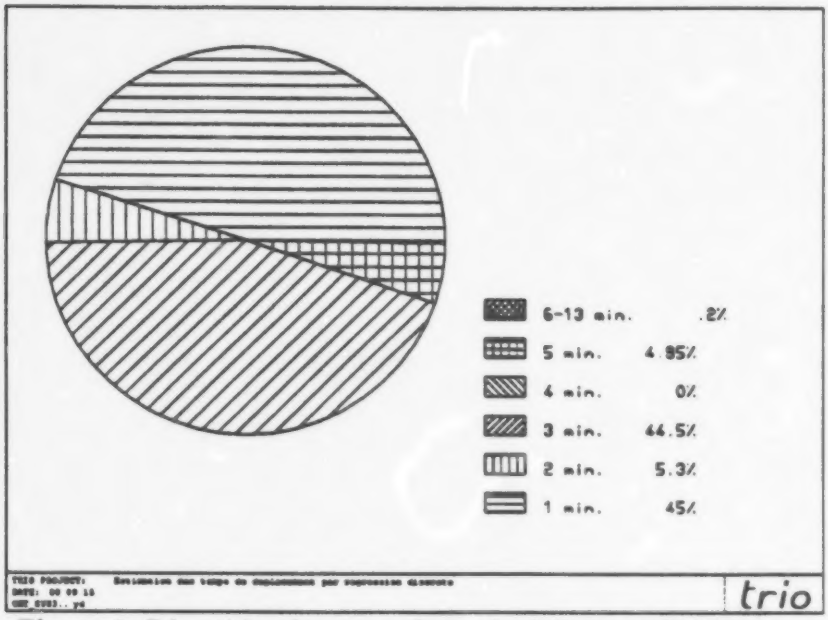


Figure 3: Répartition des temps de service à l'adresse de départ

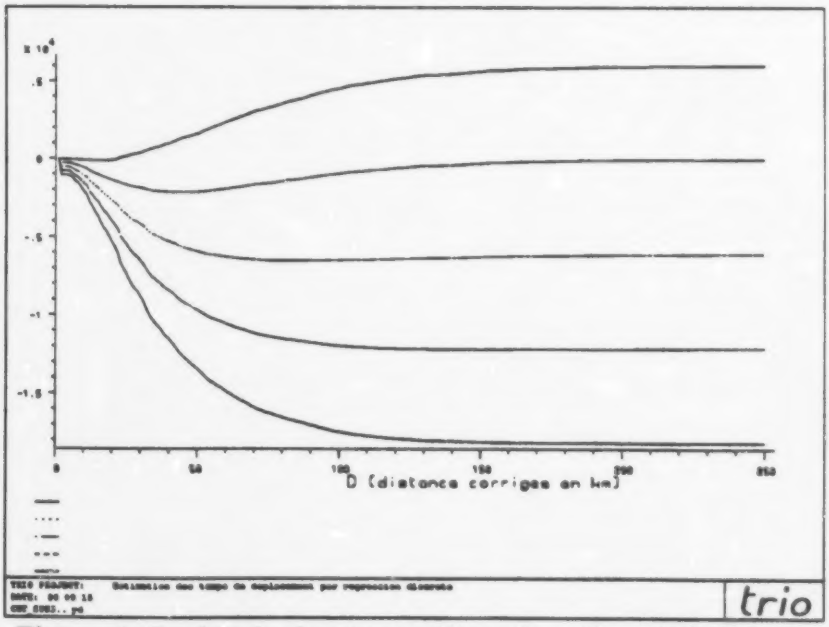


Figure 4: Courbes des fonctions unidimentionnelles $\tau(D)$ de la méthode des moindres carrés

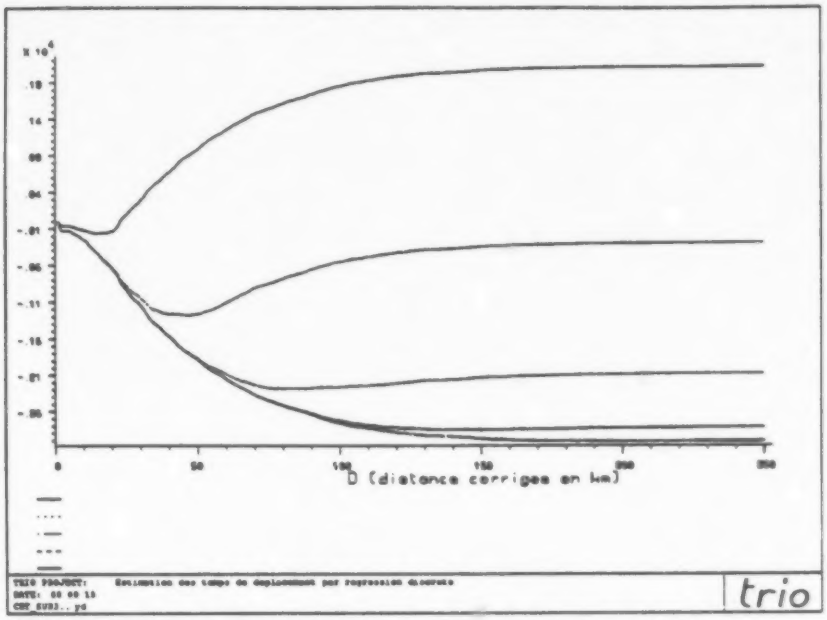


Figure 5: Courbes des fonctions unidimentionnelles $\phi(D)$ de la méthode de la somme des erreurs absolues

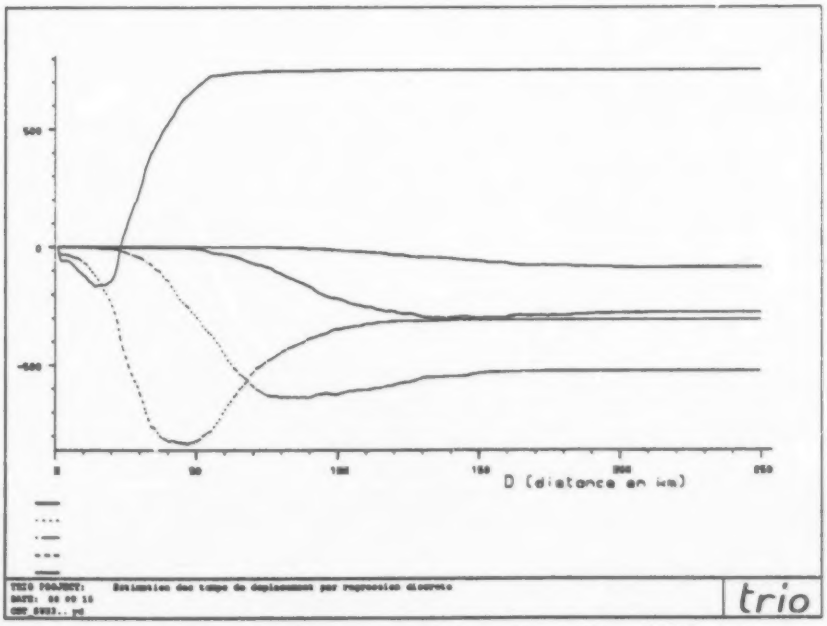


Figure 6: Courbes des fonctions unidimentionnelles $\delta(D)$ de la méthode du nombre minimum d'erreurs

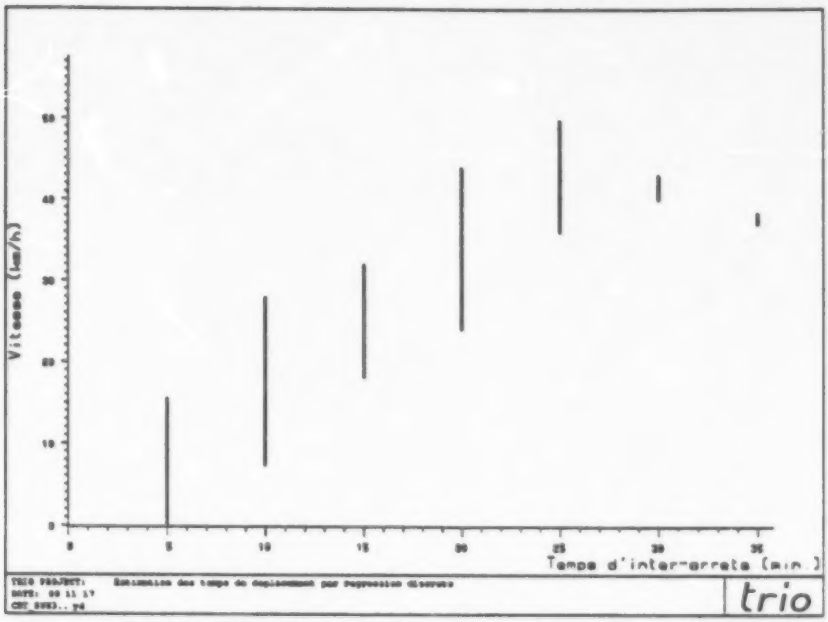


Figure 7: Intervalles de variation de la vitesse du modèle unidimentionnel pour lequel les paramètres sont estimés par le problème $(P1)$

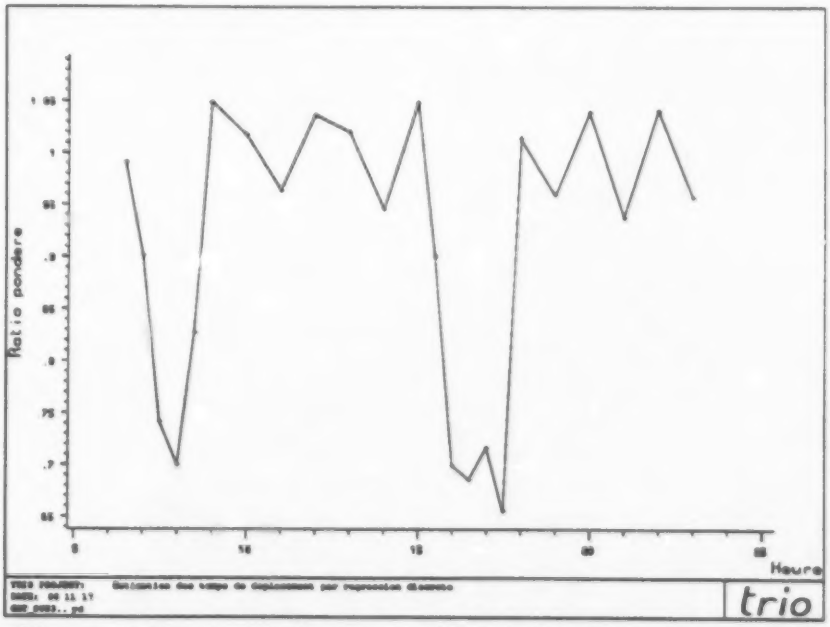


Figure 8: Courbe d'évaluation de l'effet de la période de la journée sur l'estimation des paramètres

APPROCHES	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$
moindres carrés	1.3	4.6	7.9	14.9	20.9	21.7	22.4	∞
somme minimum des erreurs	1.3	4.6	7.9	14.9	20.9	21.7	22.4	∞
nombre minimum d'erreurs	1.3	4.6	8.8	14.9	20.9	21.7	22.4	∞

Tableau 1A: Valeurs (en km) des paramètres estimés par le problème $(P1)$ pour le modèle unidimentionnel

	$T_k - \hat{T}(D_k)$								
APPROCHES	-20	-15	-10	-5	0	5	10	15	20
moindres carrés	0	2	10	419	2040	545	12	0	0
somme minimum des erreurs	0	2	10	419	2040	545	12	0	0
nombre minimum d'erreurs	0	2	9	365	2041	594	17	0	0

Tableau 1B: Distribution des erreurs $T_k - \hat{T}(D_k)$ où $\hat{T}(D)$ est définie avec les estimateurs du tableau 1A

APPROCHES	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$
moindres carrés	1.3	4.6	7.9	14.9	20.9	24.7	28.5	∞
somme minimum des erreurs	1.3	4.6	7.9	14.9	20.9	24.7	28.5	∞
nombre minimum d'erreurs	1.3	4.6	8.8	14.9	20.9	24.7	28.5	∞

Tableau 2A: Valeurs (en km) des paramètres estimés par le problème $(P2)$ pour le modèle unidimentionnel

	$T_k - \hat{T}(D_k)$								
APPROCHES	-20	-15	-10	-5	0	5	10	15	20
moindres carrés	0	2	10	419	2038	546	13	0	0
somme minimum des erreurs	0	2	10	419	2038	546	13	0	0
nombre minimum d'erreurs	0	2	9	365	2039	595	18	0	0

Tableau 2B: Distribution des erreurs $T_k - \hat{T}(D_k)$ où $\hat{T}(D)$ est définie avec les estimateurs du tableau 2A

APPROCHES	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$
moindres carrés	2.4	6.0	11.7	18.0	22.0	26.0	30.0	∞
somme minimum des erreurs	2.4	6.0	11.7	18.0	22.0	26.0	30.0	∞
nombre minimum d'erreurs	2.4	6.0	11.7	18.0	22.0	26.0	30.0	∞

Tableau 3A: Valeurs (en km) des paramètres estimés par le problème $(P2)$ pour le modèle bidimentionnel

	$T_k - \hat{T}(D_k, M_k)$								
APPROCHES	-20	-15	-10	-5	0	5	10	15	20
moindres carrés	0	1	6	201	2348	455	6	0	0
somme minimum des erreurs	0	1	6	201	2348	455	6	0	0
nombre minimum d'erreurs	0	1	6	201	2348	455	6	0	0

Note: les 11 observations aberrantes pour lesquelles $M_k \geq T_k$ ont été éliminées

Tableau 3B: Distribution des erreurs $T_k - \hat{T}(D_k, M_k)$ où $\hat{T}(D, M)$ est définie avec les estimateurs du tableau 3A

PÉRIODES	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	$\hat{\beta}_5$	$\hat{\beta}_6$	$\hat{\beta}_7$	$\hat{\beta}_8$
pointe	1.7	5.1	8.0	13.6	18.5	21.8	25.1	∞
hors pointe	2.4	6.1	12.3	18.0	22.0	26.0	30.0	∞

Note: périodes de pointe de 7:00 à 8:45 et de 15:30 à 17:45

Tableau 4A: Valeurs (en km) des paramètres estimés par le problème $(P2)$, selon la méthode des moindres carrés, pour le modèle bidimentionnel en période de pointe et en période hors pointe

	$T_k - \hat{T}(D_k, M_k)$								
PÉRIODES	-20	-15	-10	-5	0	5	10	15	20
pointe	0	1	8	204	949	102	1	0	0
hors pointe	0	1	3	89	1562	96	1	0	0
total	0	2	11	293	2511	198	2	0	0

Note: les 11 observations aberrantes pour lesquelles $M_k \geq T_k$ ont été éliminées

Tableau 4B: Distribution des erreurs $T_k - \hat{T}(D_k, M_k)$ où $\hat{T}(D, M)$ est définie avec les estimateurs du tableau 4A

A.A Appendice: démonstration des propositions

Proposition 1: Pour le modèle unidimentionnel: minimiser

$$\sum_k (e_k)^2$$

est équivalent à minimiser

$$\sum_{i=1}^{p-1} \sum_{j=1}^{p} (2j - 2i - 1) \cdot g_j(\beta_i) .$$

preuve: Pour faciliter la démonstration, on remplace l'indice k de T_k, D_k et e_k par les indices (j,k) qui réfère à la k-ième observation avec un temps de déplacement de $5j$ minutes. Soit N_j le nombre total d'observations ayant un temps d'inter-arrêts de $5j$.
Alors,

$$e_{j,k} = 5j - 5i \quad \text{si} \quad \beta_{i-1} < D_{j,k} \leq \beta_i, \quad i \in \{1, \ldots, p\}$$

$$(e_{j,k})^2 = 25 \cdot (j-i)^2 \quad \text{si} \quad \beta_{i-1} < D_{j,k} \leq \beta_i, \quad i \in \{1, \ldots, p\}$$

$$\sum_k (e_{j,k})^2 = \sum_{i=1}^{p} 25 \cdot (j-i)^2 \cdot (g_j(\beta_i) - g_j(\beta_{i-1}))$$

$$\sum_{j=1}^{p} \sum_k (e_{j,k})^2 = \sum_{j=1}^{p} \sum_{i=1}^{p} 25 \cdot (j-i)^2 \cdot (g_j(\beta_i) - g_j(\beta_{i-1}))$$

$$= 25 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p} (j-i)^2 \cdot g_j(\beta_i) - \sum_{j=1}^{p} \sum_{i=1}^{p} (j-i)^2 \cdot g_j(\beta_{i-1}) \right]]$$

$$= 25 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p-1} (j-i)^2 \cdot g_j(\beta_i) + \sum_{j=1}^{p} (j-p)^2 \cdot g_j(\beta_p) \right.$$

$$\left. - \sum_{j=1}^{p} \sum_{i=1}^{p-1} (j-i-1)^2 \cdot g_j(\beta_i) + \sum_{j=1}^{p} (j-1)^2 \cdot g_j(\beta_0) \right]$$

Puisque $g_j(\beta_p) = N_j$ et que $g_j(\beta_0) = 0$, on obtient

$$= 25 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p-1} ((j-i)^2 - (j-i-1)^2) \cdot g_j(\beta_i) + \sum_{j=1}^{p} (j-p)^2 \cdot N_j \right].$$

Puisque $(j-i)^2 - (t^2 = 2j - 2i - 1$, on obtient

$$= 25 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p-1} (2j - 2i - 1) \cdot g_j(\beta_i) + \sum_{j=1}^{p} (j-p)^2 \cdot N_j \right].$$

En éliminant les termes constants, on obtient le résultat escompté. □

Proposition 2: Pour le modèle unidimentionnel: minimiser

$$\sum_{k} |e_k|$$

est équivalent à minimiser

$$\sum_{i=1}^{p-1} \left[\sum_{j=i+1}^{p} g_j(\beta_i) - \sum_{j=1}^{i} g_j(\beta_i) \right].$$

preuve: Pour faciliter la démonstration, on remplace l'indice k de T_k, D_k et e_k par les indices (j,k) qui réfère à la k-ième observation avec un temps de déplacement de $5j$ minutes. Soit N_j le nombre total d'observations ayant un temps d'inter-arrêts de $5j$.
Alors,

$$e_{j,k} = 5j - 5i \quad \text{si} \quad \beta_{i-1} < D_{j,k} \leq \beta_i, \quad i \in \{1, \ldots, p\}$$

$$\sum_{k} |e_{j,k}| = \sum_{i=1}^{p} 5 \cdot |j-i| \cdot (g_j(\beta_i) - g_j(\beta_{i-1}))$$

$$\sum_{j=1}^{p} \sum_{k} |e_{j,k}| = \sum_{j=1}^{p} \sum_{i=1}^{p} 5 \cdot |j-i| \cdot (g_j(\beta_i) - g_j(\beta_{i-1}))$$

$$= 5 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p} |j-i| \cdot g_j(\beta_i) - \sum_{j=1}^{p} \sum_{i=1}^{p} |j-i| \cdot g_j(\beta_{i-1}) \right]$$

$$= 5 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p-1} |j-i| \cdot g_j(\beta_i) + \sum_{j=1}^{p} |j-p| \cdot g_j(\beta_p) \right.$$

$$\left. - \sum_{j=1}^{p} \sum_{i=1}^{p-1} |j-i-1| \cdot g_j(\beta_i) + \sum_{j=1}^{p} |j-1| \cdot g_j(\beta_0) \right]$$

Puisque $g_j(\beta_p) = N_j$ et que $g_j(\beta_0) = 0$, on obtient

$$= 5 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p-1} (|j-i| - |j-i-1|) \cdot g_j(\beta_i) + \sum_{j=1}^{p} |j-p| \cdot N_j \right].$$

Puisque $|j-i| - |j-i-1| = 1$ si $j > i$ et -1 sinon, on obtient

$$= 5 \cdot \left[\sum_{i=1}^{p-1} \left[\sum_{j=i+1}^{p} g_j(\beta_i) - \sum_{j=1}^{i} g_j(\beta_i) + \sum_{j=1}^{i} \cdot N_j \right] \right].$$

En éliminant les termes constants, on obtient le résultat escompté. □

Proposition 3: Pour le modèle unidimentionnel: minimiser

$$\sum_k \min(1, |e_k|)$$

est équivalent à minimiser

$$\sum_{i=1}^{p-1} \left[g_{i+1}(\beta_i) - g_i(\beta_i) \right].$$

preuve: Pour faciliter la démonstration, on remplace l'indice k de T_k, D_k et e_k par les indices (j,k) qui réfère à la k-ième observation avec un temps de déplacement de $5j$ minutes. Soit N_j le nombre total d'observations ayant un temps d'inter-arrêts de $5j$.
Alors,

$$\min(1, |e_{j,k}|) \in \{0,1\}$$

$$\sum_k \min(1, |e_{j,k}|) = g_j(\beta_{j-1}) + N_j - g_j(\beta_j)$$

$$\sum_{j=1}^{p} \sum_k \min(1, |e_{j,k}|) = \sum_{j=1}^{p} (g_j(\beta_{j-1}) + N_j - g_j(\beta_j))$$

$$= \sum_{j=1}^{p-1} g_{j+1}(\beta_j) + g_1(\beta_0) - \sum_{j=1}^{p-1} g_j(\beta_j) + g_p(\beta_p) + \sum_{j=1}^{p} N_j$$

$$= \sum_{j=1}^{p-1} \left[g_{j+1}(\beta_j) - g_j(\beta_j) \right] + g_1(\beta_0) + g_p(\beta_p) + \sum_{j=1}^{p} N_j.$$

En éliminant les termes constants, on obtient le résultat escompté. □

Proposition 4: Pour le modèle bidimentionnel: minimiser

$$\sum_k (e_k)^2$$

est équivalent à minimiser

$$\sum_{i=1}^{p-1}\sum_{j=1}^{p}\sum_{M=0}^{4}(2j-2i-1)\cdot h_j(\beta_i,M).$$

preuve: Pour faciliter la démonstration, on remplace l'indice k de D_k, M_k et e_k par les indices (j,k) qui réfère à la k-ième observation avec un temps de déplacement de $5j$ minutes. Soit N_j^M le nombre total d'observations ayant un temps de service de M et un temps d'inter-arrêts de $5j$. Alors,

$$e_{j,k} = 5j-5i \quad \text{si} \quad \beta_{i-1}\cdot\frac{5(i-1)-M'_{j,k}}{5(i-1)} < D_{j,k} \le \beta_i\cdot\frac{5i-M'_{j,k}}{5i}, \quad i\in\{1,\ldots,p\}$$

$$(e_{j,k})^2 = 25\cdot(j-i)^2 \quad \text{sous la même condition}$$

$$\sum_k (e_{j,k})^2 = \sum_{i=1}^{p} 25\cdot(j-i)^2\cdot\left[\sum_{M=0}^{4}(h_j(\beta_i,M)-h_j(\beta_{i-1},M))\right]$$

$$\sum_{j=1}^{p}\sum_k (e_{j,k})^2 = \sum_{j=1}^{p}\sum_{i=1}^{p} 25\cdot(j-i)^2\cdot\left[\sum_{M=0}^{4}(h_j(\beta_i,M)-h_j(\beta_{i-1},M))\right]$$

$$= 25\cdot\left[\sum_{j=1}^{p}\sum_{i=1}^{p}\sum_{M=0}^{4}(j-i)^2\cdot h_j(\beta_i,M) - \sum_{j=1}^{p}\sum_{i=1}^{p}\sum_{M=0}^{4}(j-i)^2\cdot h_j(\beta_{i-1},M)\right]$$

$$= 25\cdot\left[\sum_{j=1}^{p}\sum_{i=1}^{p-1}\sum_{M=0}^{4}(j-i)^2\cdot h_j(\beta_i,M) + \sum_{j=1}^{p}\sum_{M=0}^{4}(j-p)^2\cdot h_j(\beta_p,M)\right.$$
$$\left. - \sum_{j=1}^{p}\sum_{i=1}^{p-1}\sum_{M=0}^{4}(j-i-1)^2\cdot h_j(\beta_i,M) + \sum_{j=1}^{p}\sum_{M=0}^{4}(j-1)^2\cdot h_j(\beta_0,M)\right]$$

Puisque $h_j(\beta_p,M) = N_j^M$ et que $h_j(\beta_0,M) = 0$, on obtient

$$= 25\cdot\left[\sum_{j=1}^{p}\sum_{i=1}^{p-1}\sum_{M=0}^{4}((j-i)^2-(j-i-1)^2)\cdot h_j(\beta_i,M) + \sum_{j=1}^{p}\sum_{M=0}^{4}(j-p)^2\cdot N_j^M\right].$$

Puisque $(j-i)^2-(j-i-1)^2=2j-2i-1$, on obtient

$$=25\cdot\left[\sum_{j=1}^{p}\sum_{i=1}^{p-1}\sum_{M=0}^{4}(2j-2i-1)\cdot h_j(\beta_i,M)+\sum_{j=1}^{p}\sum_{M=0}^{4}(j-p)^2\cdot N_j^M\right].$$

En éliminant les termes constants, on obtient le résultat escompté. □

Proposition 5: Pour le modèle bidimentionnel, minimiser

$$\sum_k |e_k|$$

est équivalent à minimiser

$$\sum_{i=1}^{p-1}\sum_{M=0}^{4}\left[\sum_{j=i+1}^{p}h_j(\beta_i,M)-\sum_{j=1}^{i}h_j(\beta_i,M)\right].$$

preuve: Pour faciliter la démonstration, on remplace l'indice k de D_k, M_k et e_k par les indices (j,k) qui réfère à la k-ième observation avec un temps de déplacement de $5j$ minutes. Soit N_j^M le nombre total d'observations ayant un temps de service de M et un temps d'inter-arrêts de $5j$. Alors,

$$e_{j,k}=5j-5i \quad \text{si} \quad \beta_{i-1}\cdot\frac{5(i-1)-M'_{j,k}}{5(i-1)}<D_{j,k}\le\beta_i\cdot\frac{5i-M'_{j,k}}{5i}, \quad i\in\{1,\ldots,p\}$$

$$\sum_k |e_{j,k}|=\sum_{i=1}^{p}\sum_{M=0}^{4}5\cdot|j-i|\cdot(h_j(\beta_i,M)-h_j(\beta_{i-1},M))$$

$$\sum_{j=1}^{p}\sum_k |e_{j,k}|=\sum_{j=1}^{p}\sum_{i=1}^{p}\sum_{M=0}^{4}5\cdot|j-i|\cdot(h_j(\beta_i,M)-h_j(\beta_{i-1},M))$$

$$=5\cdot\left[\sum_{j=1}^{p}\sum_{i=1}^{p}\sum_{M=0}^{4}|j-i|\cdot h_j(\beta_i,M)-\sum_{j=1}^{p}\sum_{i=1}^{p}\sum_{M=0}^{4}|j-i|\cdot h_j(\beta_{i-1},M)\right]$$

$$=5\cdot\left[\sum_{j=1}^{p}\sum_{i=1}^{p-1}\sum_{M=0}^{4}|j-i|\cdot h_j(\beta_i,M)+\sum_{j=1}^{p}\sum_{M=0}^{4}|j-p|\cdot h_j(\beta_p,M)\right.$$

$$\left.-\sum_{j=1}^{p}\sum_{i=1}^{p-1}\sum_{M=0}^{4}|j-i-1|\cdot h_j(\beta_i,M)+\sum_{j=1}^{p}\sum_{M=0}^{4}|j-1|\cdot h_j(\beta_0,M)\right]$$

Puisque $h_j(\beta_p, M) = N_j^M$ et que $h_j(\beta_0, M) = 0$, on obtient

$$= 5 \cdot \left[\sum_{j=1}^{p} \sum_{i=1}^{p-1} \sum_{M=0}^{4} (|j-i| - |j-i-1|) \cdot h_j(\beta_i, M) + \sum_{j=1}^{p} \sum_{M=0}^{4} |j-p| \cdot N_j^M \right]$$

Puisque $|j-i| - |j-i-1| = 1$ si $j > i$ et -1 sinon, on obtient

$$= 5 \cdot \left[\sum_{i=1}^{p-1} \sum_{M=0}^{4} \left[\sum_{j=i+1}^{p} h_j(\beta_i, M) - \sum_{j=1}^{i} h_j(\beta_i, M) + \sum_{j=1}^{i} \cdot N_j^M \right] \right].$$

En éliminant les termes constants, on obtient le résultat escompté. □

Proposition 6: Pour le modèle bidimentionnel, minimiser

$$\sum_k \min(1, |e_k|)$$

est équivalent à minimiser

$$\sum_{i=1}^{p-1} \left[h_{i+1}(\beta_i, M) - h_j(\beta_i, M) \right].$$

preuve: Pour faciliter la démonstration, on remplace l'indice k de D_k, M_k et e_k par les indices (j,k) qui réfère à la k-ième observation avec un temps de déplacement de $5j$ minutes. Soit N_j^M le nombre total d'observations ayant un temps de service de M et un temps d'inter-arrêts de $5j$. Alors,

$$\min(1, |e_{j,k}|) \in \{0, 1\}$$

$$\sum_k \min(1, |e_{j,k}|) = \sum_{M=0}^{4} \left[h_j(\beta_{j-1}, M) + N_j^M - h_j(\beta_j, M) \right]$$

$$\sum_{j=1}^{p} \sum_k \min(1, |e_{j,k}|) = \sum_{j=1}^{p} \sum_{M=0}^{4} \left(h_j(\beta_{j-1}, M) + N_j^M - h_j(\beta_j, M) \right)$$

$$= \sum_{j=1}^{p-1} \sum_{M=0}^{4} h_{j+1}(\beta_j, M) + \sum_{M=0}^{4} h_1(\beta_0, M) - \sum_{j=1}^{p-1} \sum_{M=0}^{4} h_j(\beta_j, M) + \sum_{M=0}^{4} h_p(\beta_p, M) +$$

$$\sum_{j=1}^{p}\sum_{M=0}^{4} N_j^M$$

$$= \sum_{j=1}^{p-1}\sum_{M=0}^{4}\Big[h_{j+1}(\beta_j, M) - h_j(\beta_j, M)\Big] + \sum_{M=0}^{4} h_1(\beta_0, M) + \sum_{M=0}^{4} h_p(\beta_p, M) +$$

$$\sum_{j=1}^{p}\sum_{M=0}^{4} N_j^M.$$

En éliminant les termes constants, on obtient le résultat escompté. □

A.B Algorithme de résolution des problèmes d'estimation

A.B.1 Introduction

Nous présentons une adaptation de l'algorithme d'optimisation de l'horaire d'un itinéraire, exposé au chapitre 4, pour résoudre les problèmes d'estimation. On retrouve à la section A.B.2 une formulation plus générale du problème où les relations entre les variables sont linéaires. La section A.B.3 présente alors un nouvel algorithme de résolution. Les preuves de validité et de complexité de l'algorithme ne sont pas données puisque celles du chapitre 4 s'appliquent, à une notation près, à ce nouvel algorithme.

A.B.2. Formulation mathématique

Soient les variables $x_1, x_2, \ldots, x_n$, l'ensemble $N = \{1, 2, \ldots, n\}$ de leurs indices ainsi que les constantes $R_{i,i+1} > 0$ et $E_{i,i+1} \geq 0 \quad i = 1, \ldots, n-1$. Pour chaque variable x_i, on représente le domaine des valeurs par $[a_i, b_i]$ et le coût par la fonction convexe $f_i(x_i)$.

Le problème d'optimisation (P) est formulé de la façon suivante:

$$(P) \qquad \min \sum_{i \in N} f_i(x_i) \tag{1}$$

sujet à

$$x_i \cdot R_{i,i+1} + E_{i,i+1} \leq x_{i+1} \qquad i = 1, 2, \ldots, n-1 \tag{2}$$

$$a_i \leq x_i \leq b_i \qquad i = 1, 2, \ldots, n. \tag{3}$$

L'objectif (1) s'exprime comme une somme de fonctions convexes d'une variable, les contraintes (2) lient linéairement les variables x_i et x_{i+1} et les

contraintes (3) imposent des bornes aux valeurs des variables. Remarquons que si $R_{i,i+1} \geq 1$ pour $i = 1, \ldots, n$ alors $x_1 \leq x_2 \leq \ldots \leq x_n$. On notera par $(\bar{P})$ le problème (P) où les contraintes (2) sont relaxées.

L'étude du graphique 1 en relation avec les graphiques 4, 5 et 6 permet de voir que les fonctions $\tau(D)$, $\phi(D)$ et $\delta(D)$ sont convexes dans des intervalles $[a_i, b_i]$ contenant sûrement les valeurs optimales β_i des paramètres de la régression.

Le problème d'optimisation permettant d'estimer les paramètres de la régression s'écrit donc

$$\min \quad \sum_{i=1}^{N} f_i(\beta_i) \tag{4}$$

sujet à

$$\beta_i \cdot R_{i,i+1} \leq \beta_{i+1} \qquad i = 1, 2, \ldots, n-1 \tag{5}$$

$$a_i \leq \beta_i \leq b_i \qquad i = 1, 2, \ldots, n. \tag{6}$$

A.B.3. Algorithme

Nous présentons l'algorithme pour obtenir une solution optimale du problème (P).

L'algorithme est le suivant:

Etape 0 : (Initialisation)

- $a_{i+1} := \max\{a_{i+1},\ a_i \cdot R_{i,i+1} + E_{i,i+1}\}$ pour $i = 1, 2, \ldots, n-1$,
- $b_i := \min\{b_i,\ (b_{i+1} - E_{i,i+1})/R_{i,i+1}\}$ pour $i = n-1, n-2, \ldots, 1$,

- Soit $\bar{X} = \{\bar{x}_1, \ldots, \bar{x}_n\}$ une solution optimale du problème relaxé $(\bar{P})$,

- Initialiser $I_\emptyset := \{(1,2),(2,3),\ldots,(n-1,n)\}$,
- Poser $C := \emptyset$ et $\ell := 2$.

Etape 1 : (Test de faisabilité sur le couple (k, ℓ), où $(k, \ell) \in I_C$)

- S'il existe k tel que $(k,\ell) \in I_C$ et $\bar{x}_k \cdot R_{k,\ell} + E_{k,\ell} > \bar{x}_\ell$
  alors aller à l'étape 3
  sinon aller à l'étape 2.

Etape 2 : (Avancement d'un indice: toutes les contraintes avant l'indice ℓ sont satisfaites)

- $\ell := \ell + 1$,
- Si $\ell > n$ alors aller à l'étape 5
  sinon aller à l'étape 1.

Etape 3 : (Construction du problème réduit $(Q_{C\cup\{k\}})$)

Mise à jour de l'énoncé du problème:

- $f_\ell^{C\cup\{k\}}(x_\ell) = f_\ell^C(x_\ell) + f_k^C((x_\ell - E_{k,\ell})/R_{k,\ell})$,
- $b_\ell^{C\cup\{k\}} = b_k^C \cdot R_{k\ell} + E_{k,\ell}$,
- s'il existe j tel que $(j,k) \in I_C$
  alors $I_{C\cup\{k\}} = I_C \cup \{(j,\ell)\} - \{(j,k),(k,\ell)\}$,
  $R_{j,\ell} = R_{j,k} \cdot R_{k,\ell}$
  $E_{j,\ell} = R_{k,\ell} \cdot E_{j,k} + E_{k,\ell}$
  sinon $I_{C\cup k} = I_C - \{k,\ell\}$.

Etape 4 : (Solution optimale du problème $(\bar{Q}_{C\cup\{k\}})$)

- Soit $\bar{x}_\ell$ solution optimale du problème :

$$\min_{a_\ell \le x \le b_\ell^{C\cup\{k\}}} f_\ell^{C\cup\{k\}}(x),$$

- $C := C \cup \{k\}$,
- Aller à l'étape 1.

Etape 5 : (Solution optimale de (P))

- Pour $i = n, \ldots, 1$:

$$x_i^* = \begin{cases} \bar{x}_i & \text{si } i \in N - C \\ (x_{i+1}^* - E_{i,i+1})/R_{i,i+1} & \text{si } i \in C. \end{cases}$$

L'algorithme procède vers l'avant pour satisfaire les contraintes linéaire (2). En tout temps au cours de l'algorithme, les contraintes (2), pour $i < \ell$, sont satisfaites. En effet, à chaque itération, seule la variable x_ℓ change de valeur.

L'étape 0 effectue le pré-traitement des données et le calcul de la solution initiale du problème relaxé. L'étape 1 vérifie si la contrainte associée au couple (k, ℓ) est violée (i.e. si $\bar{x}_k \cdot R_{k,\ell} + E_{k,\ell} > \bar{x}_\ell$). Lorsque la contrainte n'est pas violée, l'étape 2 augmente ℓ pour passer à la prochaine contrainte. Lorsque la contrainte est violée le problème est réduit à l'étape 3. L'étape 4 procède à la mise à jour de la solution en réévaluant seulement $\bar{x}_\ell$. Finalement, la dernière étape de l'algorithme procède au calcul des valeurs des variables éliminées par les réductions. L'ensemble C contient l'ensemble des variables éliminées au cours de l'algorithme.